Oblivious sketching of high-degree polynomial kernels

Ahle, Thomas D.; Kapralov, Michael; Knudsen, Jakob B.T.; Pagh, Rasmus; Velingker, Ameya; Woodruff, David P.; Zandieh, Amir

Published in:
31st Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2020

Publication date:
2020

Document version
Publisher's PDF, also known as Version of record

Document license:
Unspecified

Citation for published version (APA):

Download date: 22. aug., 2022
Oblivious Sketching of High-Degree Polynomial Kernels

Thomas D. Ahle†  Michael Kapralov†  Jakob B. T. Knudsen§  Rasmus Pagh¶
Ameya Velingker‖  David P. Woodruff∗∗  Amir Zandieh††

Abstract

Kernel methods are fundamental tools in machine learning that allow detection of non-linear dependencies between data without explicitly constructing feature vectors in high dimensional spaces. A major disadvantage of kernel methods is their poor scalability: primitives such as kernel PCA or kernel ridge regression generally take prohibitively large quadratic space and (at least) quadratic time, as kernel matrices are usually dense. Some methods for speeding up kernel linear algebra are known, but they all invariably take time exponential in either the dimension of the input point set (e.g., fast multipole methods suffer from the curse of dimensionality) or in the degree of the kernel function. Oblivious sketching has emerged as a powerful approach to speeding up numerical linear algebra over the past decade, but our understanding of oblivious sketching solutions for kernel matrices has remained quite limited, suffering from the aforementioned exponential dependence on input parameters. Our main contribution is a general method for applying sketching solutions developed in numerical linear algebra over the past decade to a tensoring of data points without forming the tensoring explicitly. This leads to the first oblivious sketch for the polynomial kernel with a target dimension that is only polynomially dependent on the degree of the kernel function, as well as the first oblivious sketch for the Gaussian kernel on bounded datasets that does not suffer from an exponential dependence on the dimensionality of input data points.

1 Introduction

Data dimensionality reduction, or sketching, is a common technique for quickly reducing the size of a large-scale optimization problem while approximately preserving the solution space, thus allowing one to instead solve a much smaller optimization problem, typically in a smaller amount of time. This technique has led to near-optimal algorithms for a number of fundamental problems in numerical linear algebra and machine learning, such as least squares regression, low rank approximation, and canonical correlation analysis. In a typical instance of such a problem, one is given a large matrix $X \in \mathbb{R}^{d \times n}$ as input, and one wishes to choose a random map $\Pi$ from a certain family of random maps and replace $X$ with $\Pi X$. As $\Pi$ typically has many fewer rows than columns, $\Pi X$ compresses the original matrix $X$, which allows one to perform the original optimization problem on the much smaller matrix $\Pi X$. For a survey of such techniques, we refer the reader to the survey by Woodruff [Woo14]. A key challenge in this area is to extend sketching techniques to kernel-variants of the above linear algebra problems. Suppose each column of $X$ corresponds to an example while each of the $d$ rows corresponds to a feature. Then these algorithms require an explicit representation of $X$ to be made available to the algorithm. This is unsatisfactory in many machine learning applications, since typically the actual learning is performed in a much higher (possibly infinite) dimensional feature space, by first mapping each column of $X$ to a much higher dimensional space. Fortunately, due to the kernel trick, one need not ever perform this mapping explicitly: indeed, if the optimization problem at hand only depends on inner product information between the input points, then the kernel trick allows one to quickly compute the inner products of the high dimensional transformations of the input points, without ever explicitly computing the transformation itself. However, evaluating the kernel function easily becomes a bottleneck in algorithms that rely on the kernel trick because it typically takes $O(d)$ time to evaluate the kernel function for $d$ dimensional data. There are a number of recent works which try to improve the running times of kernel methods; we refer the reader to the recent work...
of [MM17] and the references therein. A natural question is whether it is possible to instead apply sketching techniques on the high-dimensional feature space without ever computing the high-dimensional mapping.

For the important case in which the mapping \( \phi \) is the so-called polynomial kernel, which maps \( x \in \mathbb{R}^d \) to \( \phi(x) \in \mathbb{R}^{d^p} \), where \( \phi(x)_{i_1,i_2,\ldots,i_p} = x_{i_1} x_{i_2} \cdots x_{i_p} \) for \( i_1, i_2, \ldots, i_p \in \{1, 2, \ldots, d\} \), such sketching techniques are known to be possible. This was originally shown by Pham and Pagh in the context of kernel support vector machines [PP13], using the TensorSketch technique for compressed matrix multiplication due to Pag [Pag13]. This was later extended in [ANW14] to a wide array of kernel problems in linear algebra, including principal component analysis, principal component regression, and canonical correlation analysis.

The running times of the algorithms above, while nearly linear in the number of non-zero entries of the input matrix \( X \), depend exponentially on the degree \( q \) of the polynomial kernel. For example, suppose one wishes to do low rank approximation on \( A \), the matrix obtained by replacing each column of \( X \) with its kernel-transformed version. One would like to express \( A \approx UV \), where \( U \in \mathbb{R}^{d \times k} \) and \( V \in \mathbb{R}^{k \times n} \). Writing down \( U \) explicitly is problematic, since the columns belong to the much higher \( d^p \)-dimensional space. Instead, one can express \( UV \) implicitly via column subset selection, by expressing it as a \( AZZ^\top \) and then outputting \( Z \). Here \( Z \) is an \( n \times k \) matrix. In [ANW14], an algorithm running in \( mnz(X) + (n + d)poly(3p, k, 1/\epsilon) \) time was given for outputting such \( Z \) with the guarantee that \( \|A - AZZ^\top\|_F^2 \leq (1 + \epsilon)\|A - A_k\|_F^2 \) with constant probability, where \( A_k \) is the best rank-\( k \) approximation to \( A \). Algorithms with similar running times were proposed for principal component regression and canonical correlation analysis. The main message here is that all analyses of all existing sketches require the sketch \( \Pi \) to have at least \( 3^p \) rows in order to guarantee their correctness. Moreover, the existing sketches work with constant probability only and no high probability result was known for the polynomial kernel.

The main drawback with previous work on applying dimensionality reduction for the polynomial kernel is the exponential dependence on \( p \) in the sketching dimension and consequently in the running time. Ideally, one would like a polynomial dependence. This is especially useful for the application of approximating the Gaussian kernel by a sum of polynomial kernels of various degrees, for which large values of \( p \), e.g., \( p = \text{poly}(\log n) \) are used [CKS11]. This raises the main question of our work:

**Is it possible to design a data oblivious sketch with a sketching dimension (and, hence, running time) that is not exponential in \( p \) for the above applications in the context of the polynomial kernel?**

While we answer the above question, we also study it in a more general context, namely, that of regularization. In many machine learning problems, it is crucial to regularize so as to prevent overfitting or ill-posed problems. Sketching and related sampling-based techniques have also been extensively applied in this setting. For a small sample of such work see [RR08, AM15, PW15, MM17, ACW17b, ACW17a, AKM+17]. As an example application, in ordinary least squares regression one is given a \( d \times n \) matrix \( A \), and a \( d \times 1 \) vector \( b \), and one seeks to find a \( y \in \mathbb{R}^n \) so as to minimize \( \|Ay - b\|_2^2 \). In ridge regression, we instead seek a \( y \) so as to minimize \( \|Ay - b\|_2^2 + \lambda \|y\|_2^2 \), for a parameter \( \lambda > 0 \). Intuitively, if \( \lambda \) is much larger than the operator norm \( \|A\|_2 \) of \( A \), then a good solution is obtained simply by setting \( y = 0^d \). On the other hand, if \( \lambda = 0 \), the problem just becomes an ordinary least squares regression.

In general, the statistical dimension (or effective degrees of freedom), \( s_\lambda \), captures this tradeoff, and is defined as \( \sum_{i=1}^d \lambda_i(A^\top A)^{-1} \), where \( \lambda_i(A^\top A) \) is the \( i \)-th eigenvalue of \( A^\top A \). Note that the statistical dimension is always at most \( \min(n, d) \), but in fact can be much smaller. A key example of its power is that for ridge regression, it is known [ACW17b] that if one chooses \( s_\lambda \) rows, and if \( y \) is the minimizer of \( \|\Pi Ay - \Pi b\|_2^2 + \lambda \|y\|_2^2 \), then \( \|Ay - b\|_2^2 + \lambda \|y\|_2^2 \leq (1 + \epsilon)\min_{y'}(\|Ay' - b\|_2^2 + \lambda \|y'\|_2^2) \). Note that for ordinary regression (\( \lambda = 0 \)) one would need that \( \Pi \) has \( O(\text{rank}(A)/\epsilon) \) rows [CW09]. Another drawback of existing sketches for the polynomial kernel is that their running time and target dimension depend at least quadratically on \( s_\lambda \) and no result is known with linear dependence on \( s_\lambda \), which would be optimal. We also ask if the exponential dependence on \( p \) is avoidable in the regularized setting:

**Is it possible to obtain sketching dimension bounds and running times that are not exponential in \( p \) in the context of regularization? Moreover, is it possible to obtain a running time that depends only linearly on \( s_\lambda \)?**

### 1.1 Our Contributions

In this paper, we answer the above questions in the affirmative. In other words, for each of the aforementioned applications, our algorithm depends only polynomially on \( p \). We state these applications as corollaries of our main results, which concern approximate matrix product and subspace embeddings. In particular, we devise a new distribution on oblivious linear maps \( \Pi \in \mathbb{R}^{m \times d^p} \) (i.e., a randomized family of maps that does not depend on the dataset \( X \)), so that for any fixed \( X \in \mathbb{R}^{d \times n} \), it satisfies the approx-
imate matrix product and subspace embedding properties. These are the key properties needed for kernel low
rank approximation. We remark that our data oblivious sketching is greatly advantageous to data dependent
methods because it results in a one-round distributed protocol for kernel low rank approximation \[\text{KVW14}\].

We show that our oblivious linear map \(\Pi \in \mathbb{R}^{m \times dp}\) has the following key properties:

**Oblivious Subspace Embeddings (OSEs).**

Given \(\varepsilon > 0\) and an \(n\)-dimensional subspace \(E \subseteq \mathbb{R}^d\), we say that \(\Pi \in \mathbb{R}^{m \times d}\) is an \(\varepsilon\)-subspace embedding for \(E\) if \((1 - \varepsilon)\|x\|_2 \leq \|\Pi x\|_2 \leq (1 + \varepsilon)\|x\|_2\) for all \(x \in E\).

In this paper, we focus on Oblivious Subspace Embeddings in the regularized setting. In order to define a
(regularized) Oblivious Subspace Embedding, we need to introduce the notion of statistical dimension, which
is defined as follows:

**Definition 1.1. (Statistical Dimension)**

Given \(\lambda > 0\), for every positive semidefinite matrix \(K \in \mathbb{R}^{\times n}\), we define the \(\lambda\)-statistical dimension of \(K\) to be

\[
s_\lambda(K) := tr(K(K + \lambda I_n)^{-1}).
\]

Now, we can define the notion of an oblivious subspace embedding (OSE):

**Definition 1.2. (Oblivious Subspace Embedding)**

Given \(\varepsilon, \delta, \mu > 0\) and integers \(d, n \geq 1\), an \((\varepsilon, \delta, \mu, d, n)\)-Oblivious Subspace Embedding (OSE) is a distribution \(D\) over \(m \times d\) matrices (for arbitrary \(m\)) such that for every \(\lambda > 0\), every \(A \in \mathbb{R}^{d \times n}\) with \(\lambda\)-statistical dimension \(s_\lambda(A^\top A) \leq \mu\), with probability at least \(1 - \delta\) over the randomness of the matrix \(\Pi \sim D\), the following holds\(^1\):

\[
(1.1) \quad A^\top A + \lambda I_n \preceq (\Pi A^\top I A + \lambda I_n) \preceq A^\top A + \lambda I_n.
\]

The goal is to have the target dimension \(m\) small so that \(\Pi\) provides dimensionality reduction. If we consider the non-oblivious setting where we allow the sketch matrix \(\Pi\) to depend on \(A\), then by leverage score sampling we can achieve a target dimension of \(m \approx s_\lambda(A^\top A)\), which is essentially optimal \[\text{AKM}^{+\text{18b}}\]. But as we discussed the importance of oblivious embeddings, the ultimate goal is to get an oblivious subspace embedding with target dimension of \(m \approx s_\lambda(A^\top A)\).

**Approximate Matrix Product.** We formally define this property in the following definition.

**Definition 1.3. (Approximate Matrix Product)**

Given \(\varepsilon, \delta > 0\), we say that a distribution \(D\) over \(m \times d\) matrices has the \((\varepsilon, \delta)\)-approximate matrix product property if for every \(C, D \in \mathbb{R}^{d \times n}\),

\[
\Pr_{\Pi \sim D} \left[\|C^\top \Pi \Pi A - C^\top D\|_F \leq \varepsilon \|C\|_F \|D\|_F\right] \geq 1 - \delta.
\]

Our main theorems, which provide the aforementioned guarantees, are as follows\(^2\).

**Theorem 1.1.** For every positive integers \(n, p, d\), every \(\varepsilon, s_\lambda > 0\), there exists a distribution \(\Pi^p \in \mathbb{R}^{m \times dp}\) such that: (1) If \(m = \Omega(\frac{ps_\lambda^2}{\varepsilon^2})\), then \(\Pi^p\) is an \((\varepsilon, 1/\poly(n), s_\lambda, d^p, n)\)-oblivious subspace embedding as in Definition 1.2. (2) If \(m = \tilde{\Omega}(\frac{pe^{-2}}{\varepsilon^2})\), then \(\Pi^p\) has the \((\varepsilon, 1/10)\)-approximate matrix product property as in Definition 1.3.

Moreover, for any \(X \in \mathbb{R}^{d \times n}\), if \(A \in \mathbb{R}^{dp \times n}\) is the matrix whose columns are obtained by the \(p\)-fold self-tensoring of each column of \(X\) then the matrix \(\Pi^p A\) can be computed using Algorithm 1 in time \(\tilde{O}(pnm)\).

**Theorem 1.2.** For every positive integers \(n, p, d\), every \(\varepsilon, s_\lambda > 0\), there exists a distribution \(\Pi^p \in \mathbb{R}^{m \times dp}\) such that: (1) If \(m = \tilde{\Omega}(\frac{ps_\lambda^2}{\varepsilon^2})\), then \(\Pi^p\) is an \((\varepsilon, 1/\poly(n), s_\lambda, d^p, n)\)-oblivious subspace embedding (Definition 1.2). (2) If \(m = \Omega(\frac{pe^{-2}}{\varepsilon^2})\), then \(\Pi^p\) has the \((\varepsilon, 1/\poly(n))\)-approximate matrix product property (Definition 1.3).

Moreover, in the setting of (1), for any \(X \in \mathbb{R}^{d \times n}\), if \(A \in \mathbb{R}^{dp \times n}\) is the matrix whose columns are obtained by a \(p\)-fold self-tensoring of each column of \(X\) then the matrix \(\Pi^p A\) can be computed using Algorithm 1 in time \(\tilde{O}(pnm + p^2s_\lambda^2\varepsilon^{-2}nnz(X))\).

**Theorem 1.3.** For every positive integers \(p, d, n\), every \(\varepsilon, s_\lambda > 0\), there exists a distribution \(\Pi^p \in \mathbb{R}^{m \times dp}\) which is an \((\varepsilon, 1/\poly(n), s_\lambda, d^p, n)\)-oblivious subspace embedding as in Definition 1.2 provided that the integer \(m\) satisfies \(m = \tilde{\Omega}(\frac{p^2s_\lambda^2}{\varepsilon^2})\).

Moreover, for any \(X \in \mathbb{R}^{d \times n}\), if \(A \in \mathbb{R}^{dp \times n}\) is the matrix whose columns are obtained by a \(p\)-fold self-tensoring of each column of \(X\) then the matrix \(\Pi^p A\) can be computed using Algorithm 1 in time \(\tilde{O}(pnm + p^2e^{-2}nnz(X))\).

We can immediately apply these theorems to kernel ridge regression with respect to the polynomial kernel of
degree \(p\). In this problem, we are given a regularization parameter \(\lambda > 0\), a \(d \times n\) matrix \(X\), and vector

\(^1\)For symmetric matrices \(K\) and \(K'\), the spectral inequality relation \(K \preceq K'\) holds if and only if \(x^\top Kx \leq x^\top K'x\) for all vectors \(x\).

\(^2\)Throughout this paper, the notations \(\tilde{O}, \tilde{\Omega}, \tilde{\Theta}\) suppress poly \((\log(nd/\varepsilon))\) factors.
\( b \in \mathbb{R}^n \) and would like to find a \( y \in \mathbb{R}^n \) so as to minimize \( \| A^\top Ax - b \|_2^2 + \lambda \| A^\top Ay \|_2^2 \), where \( A \in \mathbb{R}^{dp \times n} \) is the matrix obtained from \( X \) by applying the self-tensoring of degree \( p \) to each column. To solve this problem via sketching, we choose a random matrix \( \Pi \) according to the theorems above and compute \( \Pi^\top A \). We then solve the sketched ridge regression problem which seeks to minimize \( \| (\Pi^\top A)^\top \Pi^\top A x - b \|_2^2 + \lambda \| \Pi^\top A \Pi^\top A x \|_2^2 \) over \( x \). By the above theorems, we have \( \| (\Pi^\top A)^\top \Pi^\top A x - b \|_2^2 + \lambda \| \Pi^\top A \Pi^\top A x \|_2^2 = (1 \pm \varepsilon) \left( \| A^\top A x - b \|_2^2 + \lambda \| A^\top A y \|_2^2 \right) \) simultaneously for all \( x \in \mathbb{R}^n \); thus, solving the sketched ridge regression problem gives a \( (1 \pm \varepsilon) \)-approximation to the original problem. If we apply Theorem 1.1 then the number of rows of \( \Pi \) needed to ensure success with probability 9/10 is \( \Theta(p s_\lambda^2 \varepsilon^{-2}) \). The running time to compute \( \Pi^\top A \) is \( O(p^2 s_\lambda^2 \varepsilon^{-2} n + p \minz(X)) \), after which a ridge regression problem can be solved in \( O(n s_\lambda^2 / \varepsilon^4) \) time via an exact closed-form solution for ridge regression. An alternative approach to obtaining a very high-accuracy approximation is to use the sketched kernel as a preconditioner to solve the original ridge regression problem, which improves the dependence on \( \varepsilon \) to \( \log(1/\varepsilon) \) [ACW17a]. To obtain a higher probability of success, we can instead apply Theorem 1.3 which would allow us to compute the sketched matrix \( \Pi^\top A \) in \( O(p^5 s_\lambda \varepsilon^{-2} n + p^3 \varepsilon^{-2} \minz(X)) \) time. This is the first sketch to achieve the optimal dependence on \( s_\lambda \) for the polynomial kernel, after which we can now solve the ridge regression problem in \( O(n s_\lambda^2 \text{poly}(p, \varepsilon^{-1})) \) time. Importantly, both running times are polynomial in \( p \), whereas all previously known methods incurred running times that were exponential in \( p \).

Although there has been much work on sketching methods for kernel approximation which nearly achieve the optimal target dimension \( m \approx s_\lambda \), such as Nyström sampling [MM17], all known methods are data-dependent unless strong conditions are assumed about the kernel matrix (small condition number or incoherence). Data oblivious methods provide nice advantages, such as one-round distributed protocols and single-pass streaming algorithms. However, for kernel methods they are poorly understood and previously had worse theoretical guarantees than data-dependent methods. Furthermore, note that the Nyström method requires to sample at least \( m = \Omega(s_\lambda) \) landmarks to satisfy the subspace embedding property even given an oracle access to the exact leverage scores distribution. This results in a runtime of \( \Omega(s_\lambda^2 d + s_\lambda \minz(X)) \). Whereas our method achieves a target dimension that nearly matches the best dimension possible with data-dependent Nyström method and with strictly better running time of \( O(ns_\lambda + \minz(X)) \) time (assuming \( p = \text{poly} (\log n) \)). Therefore, for a large range of parameter our sketch runs in input sparsity time whereas the Nyström methods are slower by an \( s_\lambda \) factor in the best case.

**Application: Polynomial Kernel Rank-k Approximation.** Approximate matrix product and subspace embedding are key properties for sketch matrices which imply efficient algorithms for rank-k kernel approximation [ANW14]. The following corollary of Theorem 1.4 immediately follows from Theorem 6 of [ANW14].

**Corollary 1.1. (Rank-k Approximation)** For every positive integers \( k, n, p, d \), every \( \varepsilon > 0 \), any \( X \in \mathbb{R}^{d \times n} \), if \( A \in \mathbb{R}^{d \times n} \) is the matrix whose columns are obtained by the \( p \)-fold self-tensoring of each column of \( X \) then there exists an algorithm which finds an \( n \times k \) matrix \( V \) in time \( O(p \minz(X) + \text{poly}(k, p, \varepsilon^{-1})) \) such that with probability 9/10,

\[
\| A - AVV^\top \|_F^2 \leq (1 + \varepsilon) \min_{U \in \mathbb{R}^{d \times n}} \| A - U \|_F^2.
\]

Note that this runtime improves the runtime of [ANW14] by exponential factors in the polynomial kernel’s degree \( p \).

**Additional Applications.** Our results also imply improved bounds for each of the applications in [ANW14], including canonical correlation analysis (CCA), and principal component regression (PCR). Importantly, we obtain the first sketching-based solutions for these problems with running time polynomial rather than exponential in \( p \).

**Oblivious Subspace Embedding for the Gaussian Kernel.** One very important implication of our result is Oblivious Subspace Embedding of the Gaussian kernel. Most work in this area is related to the Random Fourier Features method [RR08]. It was shown in [AKM17] that one requires \( \Omega(n) \) samples of the standard Random Fourier Features to obtain a subspace embedding for the Gaussian kernel, while a modified distribution for sampling frequencies yields provably better performance. The target dimension of our proposed sketch for the Gaussian kernel strictly improves upon the result of [AKM17], which has an exponential dependence on the dimension \( d \). We for the first time, embed the Gaussian kernel with a target dimension which has a linear dependence on the statistical dimension of the kernel and is not exponential in the dimensionality of the data-point.

**Theorem 1.4.** For every \( r > 0 \), every positive integers \( n, d, \) and every \( X \in \mathbb{R}^{d \times n} \) such that \( \| x_i \|_2 \leq r \) for all \( i \in [n] \), where \( x_i \) is the \( i \)-th column of \( X \), suppose \( G \in \mathbb{R}^{n \times n} \) is the Gaussian kernel matrix –
i.e., \( G_{i,k} = e^{-\|x_j - x_k\|^2/2} \) for all \( j, k \in [n] \). There exists an algorithm which computes \( S_g(X) \in \mathbb{R}^{m \times n} \) in time \( \tilde{O}(q^6 \varepsilon^{-2} n s \lambda + q^6 \varepsilon^{-2} \text{nnz}(X)) \) such that for every \( \varepsilon, \lambda > 0 \), the following holds with probability at least \( 1 - \frac{1}{\text{poly}(n)} \),

\[
G + \lambda I_n 
\leq \left( S_g(X) \right)^\top S_g(X) + \lambda I_n 
\leq G + \lambda I_n
\]

where \( m = \Theta(q^5 s \lambda / \varepsilon^2) \) and \( q = \Theta(r^2 + \log(n/\varepsilon \lambda)) \) and \( s \lambda \) is \( \lambda \)-statistical dimension of \( G \) as in Definition 1.1.

We remark that for datasets with radius \( r = \text{poly}(\log n) \) even if one has oracle access to the exact leverage scores for Fourier features of Gaussian kernel, in order to get subspace embedding guarantee one needs to use \( m = \Omega(s \lambda) \) features which requires \( \Omega(s \lambda \text{nnz}(X)) \) operations to compute. Whereas our result of Theorem 1.4 runs in time \( \tilde{O}(n s \lambda + \text{nnz}(X)) \). Therefore, for a large range of parameters our Gaussian sketch runs in input sparsity time whereas the Fourier features method is at best slower by an \( s \lambda \) factor.

### 1.2 Technical Overview

Our goal is to design a sketching matrix \( \Pi^p \) that satisfies the oblivious subspace embedding property with an optimal embedding dimension and which can be efficiently applied to vectors of the form \( x^{\otimes p} \in \mathbb{R}^d \). We start by describing some natural approaches to this problem (some of which have been used before), and show why they incur an exponential loss in the degree of the polynomial kernel. We then present our sketch and outline our proof of its correctness.

We first discuss two natural approaches to tensoring classical sketches, namely the Johnson-Lindenstrauss transform and the CountSketch. We show that both lead to an exponential dependence of the target dimension on \( p \) and then present our new approach.

**Tensoring the Johnson-Lindenstrauss Transform.** Perhaps the most natural approach to designing a sketch \( \Pi^p \) is the idea of tensoring \( p \) independent Johnson-Lindenstrauss matrices. Specifically, let \( m \) be the target dimension. For every \( r = 1, \ldots, p \) let \( M^{(r)} \) denote an \( m \times d \) matrix with iid uniformly random \( \pm 1 \) entries, and let the sketching matrix \( M \in \mathbb{R}^{m \times d^p} \) be

\[
M = \frac{1}{\sqrt{m}} M^{(1)} \bullet \ldots \bullet M^{(p)},
\]

where \( \bullet \) stands for the operation of tensoring the rows of matrices \( M^{(r)} \) (see Definition 2.4). This would be a very efficient matrix to apply, since for every \( j = 1, \ldots, m \)

the \( j \)-th entry of \( M x^{\otimes p} \) is exactly \( \prod_{r=1}^p [M^{(r)} x]_j \), which can be computed in time \( O(p \text{nnz}(x)) \), giving overall evaluation time \( O(p m \text{nnz}(x)) \). One would hope that \( m = \Omega(\varepsilon^{-2} \log n) \) would suffice to ensure that \( \|M x^{\otimes p}\|^2 = (1 \pm \varepsilon)\|x^{\otimes p}\|^2 \). However, this is not true: we show in full version of this paper that one must have \( m = \Omega(\varepsilon^{-2} p \log(n)/p + \varepsilon^{-1} (\log(n)/p)^p) \) in order to preserve the norm with high probability. Thus, the dependence on degree \( p \) of the polynomial kernel must be exponential. The lower bound is provided by controlling the moments of the sketch \( M \) and using Paley-Zygmund inequality. For completeness, we show that the aforementioned bound on the target dimension \( m \) is sharp, i.e., necessary and sufficient for obtaining the Johnson-Lindenstrauss property.

**Tensoring of CountSketch (TensorSketch).** Pagh and Pham [Pagh and Pham, 2013] introduced the following tensorized version of CountSketch. For every \( i = 1, \ldots, p \) let \( h_i : [d] \rightarrow [m] \) denote a random hash function, and \( \sigma_i : [d] \rightarrow [m] \) a random sign function. Then let \( S : \mathbb{R}^{d \otimes p} \rightarrow \mathbb{R}^m \) be defined by

\[
S_{r,(j_1, \ldots, j_p)} := \sigma(i_1) \cdots \sigma(i_p) 1[h_1(i_1) + \ldots + h_p(i_p) = r]
\]

for \( r = 1, \ldots, m \). For every \( x \in \mathbb{R}^d \) one can compute \( S x^{\otimes p} \) in time \( O(p m \log m + p \text{nnz}(x)) \). Since the time to apply the sketch only depends linearly on the dimension \( p \) (due to the Fast Fourier Transform) one might hope that the dependence of the sketching dimension on \( p \) is polynomial. However, this turns out to not be the case: the argument in [Sonar and Wooters, 2014] implies that \( m = \tilde{O}(3^p s \lambda^2) \) suffices to construct a subspace embedding for a matrix with regularization \( \lambda \) and statistical dimension \( s \lambda \), and we show in full version of this work that exponential dependence on \( p \) is necessary.

**Our Approach: Recursive Tensoring.** The initial idea behind our sketch is as follows. To apply our sketch \( \Pi^p \) to \( x^{\otimes p} \), for \( x \in \mathbb{R}^d \), we first compute the sketches \( T_1 x, T_2 x, \ldots, T_p x \) for independent sketching matrices \( T_1, \ldots, T_p \sim T_{\text{base}} \) – see the leaves of the sketching tree in Fig. 1 Note that we choose these sketches as CountSketch or OSNAP to ensure that the leaf sketches can be applied in time proportional to the number of nonzeros in the input data (in the case of OSNAP this is true up to polylogarithmic factors).

Each of these is a standard sketching matrix mapping \( d \)-dimensional vectors to \( m \)-dimensional vectors for some common value of \( m \). We refer the reader to the survey [Wooters, 2014]. The next idea is to choose new sketching matrices \( S_1, S_2, \ldots, S_{p/2} \sim S_{\text{base}} \) mapping \( m^2 \)-dimensional vectors to \( m \)-dimensional vectors and apply \( S_1 \) to \( (T_1 x) \otimes (T_2 x) \), as well as apply \( S_2 \) to \( (T_3 x) \otimes (T_4 x) \), and so on, applying \( S_{p/2} \) to \( (T_{p-1} x) \otimes (T_p x) \).
Figure 1: $S_{\text{base}}$ is chosen from the family of sketches which support fast matrix-vector product for tensor inputs such as TensorSketch and TensorSRHT. The $T_{\text{base}}$ is chosen from the family of sketches which operate in input sparsity time such as CountSketch and OSNAP.

These sketches are denoted by $S_{\text{base}}$ – see internal nodes of the sketching tree in Fig. 1. We note that in order to ensure efficiency of our construction (in particular, running time that depends only linearly on the statistical dimension $s_\lambda$) we must choose $S_{\text{base}}$ as a sketch that can be computed on tensored data without explicitly constructing the actual tensored input, i.e., $S_{\text{base}}$ supports fast matrix vector product on tensor product of vectors. We use either TensorSketch (for results that work with constant probability) and a new variant of the Subsampled Randomized Hadamard Transform SRHT which supports fast multiplication for the tensoring of two vectors (for high probability bounds) – we call the last sketch TensorSRHT.

At this point we have reduced our number of input vectors from $p$ to $p/2$, and the dimension is $m$, which will turn out to be roughly $s_\lambda$. We have made progress, as we now have fewer vectors each in roughly the same dimension we started with. After log₂$p$ levels in the tree we are left with a single output vector.

Intuitively, the reason that this construction avoids an exponential dependence on $p$ is that at every level in the tree we use target dimension $m$ larger than the statistical dimension of our matrix by a factor polynomial in $p$. This ensures that the accumulation of error is limited, as the total number of nodes in the tree is $O(p)$. This is in contrast to the direct approaches discussed above, which use a rather direct tensoring of classical sketches, thereby incurring an exponential dependence on $p$ due to dependencies that arise.

**Showing Our Sketch is a Subspace Embedding.** In order to show that our recursive sketch is a subspace embedding, we need to argue it preserves norms of arbitrary vectors in $\mathbb{R}^{d^p}$, not only vectors of the form $x^{\otimes p}$. Indeed, all known methods for showing the subspace embedding property (see [Woo14] for a survey) at the very least argue that the norms of each of the columns of an orthonormal basis for the subspace in question are preserved. While our subspace may be formed by the span of vectors which are tensor products of $p$ $d$-dimensional vectors, we are not guaranteed that there is an orthonormal basis of this form. Thus, we first observe that our mapping is indeed linear over $\mathbb{R}^{d^p}$, making it well-defined on the elements of any basis for our subspace, and hence our task essentially reduces to proving that our mapping preserves norms of arbitrary vectors in $\mathbb{R}^{d^p}$.

We present two approaches to analyzing our construction. One is based on the idea of propagating moment bounds through the sketching tree, and results in a nearly linear dependence of the sketching dimension $m$ on the degree $p$ of the polynomial kernel, at the expense of a quadratic dependence on the statistical dimension $s_\lambda$. This approach is presented in Section 3. The other approach achieves the (optimal) linear dependence on $s_\lambda$, albeit at the expense of a worse polynomial dependence on $p$. This approach uses sketches that succeed with high probability, and uses matrix concentration bounds.

**Propagating moment bounds through the tree – optimizing the dependence on the degree $p$.** We analyze our recursively tensored version of the OSNAP and CountSketch by showing how moment bounds can be propagated through the tree structure of the sketch. This analysis is presented in Section 4 and results in the proof of Theorem 1.1 as well as the first part of Theorem 1.3. The analysis obtained this way give particularly sharp dependencies on $p$ and log $1/\delta$.

The idea is to consider the unique matrix $M \in \mathbb{R}^{m \times d^p}$ that acts on simple tensors in the way we have described it recursively above. This matrix could in principle be applied to any vector $x \in \mathbb{R}^{d^p}$ (though it would be slow to realise). We can nevertheless show that this matrix has the $(\varepsilon, \delta, t)$-JL Moment Property, which is for parameters $\varepsilon, \delta \in [0, 1], t \geq 2$, and every $x \in \mathbb{R}^{d}$ with $\|x\|_2 = 1$ the statement $\mathbb{E}\left[\|Mx\|_2^t - 1\right] \leq \varepsilon^t \delta$.

It can be shown that $M$ is built from our various $S_{\text{base}}$ and $T_{\text{base}}$ matrices using three different operations: multiplication, direct sum, and row-wise tensoring. In other words, it is sufficient to show that if $Q$ and $Q'$ both have the $(\varepsilon, \delta, t)$-JL Moment Property, then so does $QQ'$, $Q \oplus Q'$ and $Q \bullet Q'$. This turns out to hold for $Q \oplus Q'$, but $QQ'$ and $Q \bullet Q'$ are more tricky. (Here $\oplus$ and $\bullet$ denote $x \otimes x \cdots \otimes x$, the $p$-fold self-tensoring of $x$. \[4\text{x^{\otimes p}}\] denotes $x \otimes x \cdots \otimes x$, the $p$-fold self-tensoring of $x$. \[\text{p terms}\]}
is the direct sum and \(\circ\) is the composition of tensoring the rows. See section 2 on notation.)

For multiplication, a simple union bound allows us to show that \(Q^{(1)} Q^{(2)} \cdots Q^{(p)}\) has the \((p,\varepsilon, p\delta, t)\)-JL Moment Property. This would unfortunately mean a factor of \(p^2\) in the final dimension. The union bound is clearly suboptimal, since implicitly it is assumes that all the matrices conspire to either shrink or increase the norm of a vector, while in reality with independent matrices, we should get a random walk on the real line. Using an intricate decoupling argument, we show that this is indeed the case, and that \(Q^{(1)} Q^{(2)} \cdots Q^{(p)}\) has the \((\sqrt{p} \varepsilon, \delta, t)\)-JL Moment Property, saving a factor of \(p\) in the output dimension.

Finally we need to analyze \(Q \odot Q'\). Here it is easy to show that the JL Moment Property doesn’t in general propagate to \(Q \odot Q’\) (consider e.g. \(Q\) being constant 0 on its first \(m/2\) rows and \(Q’\) having 0 on its \(m/2\) last rows.) For most known constructions of JL matrices it does however turn out that \(Q \odot Q’\) behaves well. In particular we show this for matrices with independent sub-Gaussian entries, and for the so-called Fast Johnson Lindenstrauss construction (lemma 4.11). The main tool here is a higher order version of the classical Khintchine inequality [HM07], which bounds the moments \(\mathbb{E}[(\sigma^{(1)} \odot \sigma^{(2)} \odot \cdots \odot \sigma^{(p)}, x)']\) when \(\sigma^{(1)}, \ldots, \sigma^{(p)}\) are independent sub-Gaussian vectors (lemma 4.9).

Optimizing the dependence on \(s\lambda\). Our proof of Theorem 1.3 relies on instantiating our framework with OSNAP at the leaves of the tree (\(T_{base}\)) and a novel version of the SRHT that we refer to as TensorSRHT at the internal nodes of the tree. We outline the analysis here. In order to show that our sketch preserves norms, let \(y\) be an arbitrary vector in \(\mathbb{R}^{d^p}\). Then in the bottom level of the tree, we can view our sketch as \(T_1 \times T_2 \times \cdots \times T_p\), where \(x\) for denotes the tensor product of matrices (see Definition 2.2). Then, we can reshape \(y\) to be a \(d^{p-1} \times d\) matrix \(Y\), and the entries of \(T_1 \times T_2 \times \cdots \times T_p y\) are in bijective correspondence with those of \(T_1 \times T_2 \times \cdots \times T_{p-1} Y T_p^\top\). By definition of \(T_p\), it preserves the Frobenius norm of \(Y\), and consequently, we can replace \(Y\) with \(Y T_p^\top\). We next look at \((T_1 \times T_2 \times \cdots \times T_{p-1}) Z (I_d \times T_{p-1}^\top)\), where \(Z\) is the \(d^{p-2} \times d^2\) matrix with entries in bijective correspondence with those of \(Y T_p^\top\). Then we know that \(T_{p-1}\) preserves the Frobenius norm of \(Z\). Iterating in this fashion, this means the first layer of our tree preserves the norm of \(y\), provided we union bound over \(O(p)\) events that a sketch preserves a norm of an intermediate matrix. The core of the analysis consists of applying spectral concentration bounds based analysis to sketches that act on blocks of the input vector in a correlated fashion. We give the details in Section 5.

**Sketching the Gaussian kernel.** Our techniques yield the first oblivious sketching method for the Gaussian kernel with target dimension that does not depend exponentially on the dimensionality of the input data points. The main idea is to Taylor expand the Gaussian function and apply our sketch for the polynomial kernel to the elements of the expansion. It is crucial here that the target dimension of our sketch for the polynomial kernel depends only polynomially on the degree, as otherwise we would not be able to truncate the Taylor expansion sufficiently far in the tail (the number of terms in the Taylor expansion depends on the radius of the dataset and depends logarithmically on the regularization parameter). Overall, our Gaussian kernel sketch has optimal target dimension up to polynomial factors in the radius dataset and logarithmic factors in the dataset size. Moreover, it is the first subspace embedding of Gaussian kernel which runs in input sparsity time \(O(\text{nnz}(X))\) for datasets with polylogarithmic radius. The result is summarized in Theorem 1.4 and the analysis is presented in Section 6.

### 1.3 Related Work

Work related to sketching of tensors and explicit kernel embeddings is found in fields ranging from pure mathematics to physics and machine learning.

**Johnson-Lindenstrauss Transform** A cornerstone result in the field of subspace embeddings is the Johnson-Lindenstrauss lemma [JLS86], which is known to give an \(r\)-dimensional subspace embedding with a taget dimension of \(m = O(\varepsilon^{-2} r)\) [CW13, CNW16]. Achlioptas [Ach03] constructed a JL transform which uses \(O(\text{nnz}(x))\) runtime to sketch \(x \in \mathbb{R}^d\). Later, the Fast Johnson Lindenstrauss Transform [AC06] improved the running time to \(O(d \log d + m^3)\). The related Subsampled Randomized Hadamard Transform (SRHT) has been extensively studied [Sar06, DMM06, DMMS11, Tyr11, DMW12, LDFU13], which uses \(O(d \log d)\) time with a suboptimal \(m \approx \varepsilon^{-2} \log(1/\delta)^2\).

The above improvements have a running time of \(O(d \log d)\), which is unsatisfactory for sparse inputs. This inspired a line of work trying to obtain sparse JL transforms [DKS10, KN14, NN13, Coh16], which runs in time \(O(\varepsilon^{-1} \log(1/\delta) \text{nnz}(x))\). NN13 introduced ONSAP transform and showed a trade-off between sparsity and embedding dimension. This was further improved in [Coh16].

In the context of this paper all the above mentioned methods have the same shortcoming, they do not exploit the structure of the tensors. The SRHT has a running time of \(\Omega(pd^p \log d)\) in our model, and the sparse embeddings have a running time of \(\Omega(\text{nnz}(x)p)\)
This is clearly unsatisfactory and inspired the TensorSketch \[\text{[PP13][Woo14]}, \text{which has a running time of } \Omega(p \log n(x)), \text{but unfortunately, needs } m = \Omega(3^2 \epsilon^{-2} \delta^{-1}) \text{with exponential dependence on } p. \]

Approximate Kernel Expansions A classic result by Rahimi and Recht [RR08] shows how to compute an embedding for any shift-invariant kernel function \(k\) with exponential dependence on \(m\). \[\text{[LSS14]}\] extended this to any kernel on the form \(k(x, y)\) using time \(O((m + d) \log d)\), however the method does not handle kernel functions that can’t be specified as a function of the inner product, and it doesn’t provide subspace embeddings. See also [MM17] for more approaches along these lines. Unfortunately, these methods are unable to operate in input sparsity time and their runtime is at best off by an \(s\) factor.

Tensor Sparsification and Hyper-plane rounding There is a literature of tensor sparsification based on sampling [NDT15], however unless the tensored vectors are very smooth (such as \(±1\) vectors), the sampling has to be weighted by the data which makes it inapplicable to the data oblivious problems we consider. An alternative approach is to use hyper-plane rounding to get vectors on the set \(±1\). This method was first brought into the field of data-analysis by [Cha02] and [Val15] was the first to use it with tensoring. The main issue with this approach is that it isn’t a linear sketch, which hinders the applications we consider in this paper, such as low rank approximation, CCA, PCR, ridge regression. It also takes \(dm\) time to calculate \(Mx\) and \(My\) which is unsatisfactory.

1.4 Organization In section 2 we introduce basic definitions and notations. Section 3 introduces our recursive construction of the sketch for tensorn products. Section 4 analyzes how the moment bounds propagate through our recursive construction thereby proving Theorems 1.1 and 1.2. Section 5 introduces a high probability Oblivious Subspace Embedding with linear dependence on the statistical dimension thereby proving Theorem 1.3. Finally, section 6 uses the tools that we build for sketching polynomial kernel and proves that, for the first time, Gaussian kernel can be sketched without an exponential loss in the dimension.

2 Preliminaries
In this section we introduce notation and basic definitions. We denote the tensor product of vectors \(a, b\) by \(a \otimes b\) which is formally defined as follows.

**Definition 2.1. (Tensor Product of Vectors)**
Given \(a \in \mathbb{R}^m\) and \(b \in \mathbb{R}^n\) we define the twofold tensor product \(a \otimes b\) to be the vector \((a_1 b_1, a_2 b_1, \ldots, a_m b_1, a_1 b_2, a_2 b_2, \ldots, a_m b_2, \ldots, a_m b_n)\).

Given \(v_1 \in \mathbb{R}^d_1, v_2 \in \mathbb{R}^d_2, \ldots v_k \in \mathbb{R}^d_k\), we define the \(k\)-fold tensor product \(v_1 \otimes v_2 \cdots \otimes v_k \in \mathbb{R}^{d_1 \cdots d_k}\). For shorthand, we use the notation \(v^\otimes k\) to denote \(v \otimes v \cdots \otimes v\), the \(k\)-fold self-tensoring of \(v\).

Tensor product can be naturally extended to matrices,

**Claim 1.** For every matrices \(A, B, C, D\) with appropriate sizes, \((A \cdot B) \times (C \cdot D) = (A \times C) \cdot (B \times D)\).

The column wise tensoring of matrices is defined as,

**Definition 2.2.** Given \(A_1 \in \mathbb{R}^{m_1 \times n_1}, A_2 \in \mathbb{R}^{m_2 \times n_2}, \ldots, A_k \in \mathbb{R}^{m_k \times n_k}\), we define \(A_1 \otimes A_2 \otimes \cdots \otimes A_k\) to be the matrix in \(\mathbb{R}^{m_1 m_2 \cdots m_k n_1 n_2 \cdots n_k}\) whose element at row \((i_1, \ldots, i_k)\) and column \((j_1, \ldots, j_k)\) is \(A_1(i_1, j_1) \cdots A_k(i_k, j_k)\). As a consequence the following holds for any \(v_1 \in \mathbb{R}^{n_1}, v_2 \in \mathbb{R}^{n_2}, \ldots, v_k \in \mathbb{R}^{n_k}:

\[(A_1 \times A_2 \times \cdots \times A_k)(v_1 \otimes v_2 \cdots \otimes v_k) = (A_1 v_1) \otimes (A_2 v_2) \cdots \otimes (A_k v_k)\]

Similarly the row wise tensoring of matrices are introduced in the following Definition,

**Definition 2.3.** Given \(A_1 \in \mathbb{R}^{m_1 \times n_1}, A_2 \in \mathbb{R}^{m_2 \times n_2}, \ldots, A_k \in \mathbb{R}^{m_k \times n_k}\), we define \(A_1 \otimes A_2 \otimes \cdots \otimes A_k\) to be the matrix in \(\mathbb{R}^{m_1 \cdot m_2 \cdots m_k \times n_1 \cdot n_2 \cdots n_k}\) whose \(j\)th column is \(A_1^j \otimes A_2^j \otimes \cdots \otimes A_k^j\) for every \(j \in [n]\), where \(A_1^j\) is the \(j\)th column of \(A_1\) for every \(l \in [k]\).

Another related operation is the direct sum for vectors: \(x \oplus y = [y \downarrow x]\) and for matrices: \(A \oplus B = \begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}\).

When the sizes match up, we have \((A \oplus B)(x \oplus y) = Ax + By\). Also notice that if \(I_k\) is the \(k \times k\) identity matrix, then \(I_k \otimes A = A \oplus \cdots \oplus A\).

3 Construction of the Sketch
In this section, we present the construction for our new sketch. Suppose we are given \(v_1, v_2, \ldots, v_q \in \mathbb{R}^m\). Our main task is to map the tensor product \(v_1 \otimes v_2 \otimes \cdots \otimes v_q\) to a vector of size \(m\) using a linear sketch. Our sketch construction is recursive in nature. To illustrate the general idea, let us first consider the case in which \(q \geq 2\) is a power of two. Our sketch involves first
sketching each pair \((v_1 \otimes v_2), (v_3 \otimes v_4), \ldots, (v_{q-1} \otimes v_q) \in \mathbb{R}^{m^2}\) independently using independent instances of some linear base sketch (e.g., degree two TensorSketch, Subsampled Randomized Hadamard Transform (SRHT), CountSketch, OSNAP). The number of vectors after this step is half of the number of vectors that we began with. The natural idea is to recursively apply the same procedure on the sketched tensors with half as many instances of the base sketch in each successive step.

More precisely, we first choose a (randomized) base sketch \(S_{\text{base}} : \mathbb{R}^{m_2} \rightarrow \mathbb{R}^m\) that sketches twofold tensor products of vectors in \(\mathbb{R}^m\) (we will describe how to choose the base sketch later). Then, for any power of two \(q \geq 2\), we define \(Q^q : \mathbb{R}^{m^q} \rightarrow \mathbb{R}^m\) on \(v_1 \otimes v_2 \otimes \cdots \otimes v_q\) recursively as follows:

\[
Q^q(v_1 \otimes \cdots \otimes v_q) = Q^2 \left( S^2_1(v_1 \otimes v_2) \otimes \cdots \otimes S^2_{q/2}(v_{q-1} \otimes v_q) \right),
\]

where \(S^2_1, S^2_2, \ldots, S^2_{q/2} : \mathbb{R}^{m^2} \rightarrow \mathbb{R}^m\) are independent instances of \(S_{\text{base}}\) and \(Q^1 : \mathbb{R}^m \rightarrow \mathbb{R}^m\) is simply the identity map on \(\mathbb{R}^m\).

The above construction of \(Q^q\) is defined in terms of its action on \(q\)-fold tensor products of vectors in \(\mathbb{R}^m\), but it extends naturally to a linear mapping from \(\mathbb{R}^{m^q}\) to \(\mathbb{R}^m\). The formal definition of \(\Pi^q\) is presented below.

**Definition 3.1. (Sketch \(Q^q\))** Let \(m \geq 2\) be a positive integer and let \(S_{\text{base}} : \mathbb{R}^{m_2} \rightarrow \mathbb{R}^m\) be a linear map that specifies some base sketch. Then, for any integer power of two \(q \geq 2\), we define \(Q^q : \mathbb{R}^{m^q} \rightarrow \mathbb{R}^m\) to be the linear map specified as \(Q^q \equiv S^2 \cdot S^4 \cdot \cdots \cdot S^q/2 \cdot S^q\), where for each \(l \in \{2, 4, \ldots, q/2, q\}\), \(S^l\) is a matrix in \(\mathbb{R}^{m^{l/2} \times m^{l}}\) defined as

\[
S^l \equiv S^l_1 \times S^l_2 \times \cdots \times S^l_{l/2},
\]

where the matrices \(S^l_1, \ldots, S^l_{l/2} \in \mathbb{R}^{m^{l/2} \times m^{l/2}}\) are drawn independently from a base distribution \(S_{\text{base}}\).

This sketch construction can be best visualized using a balanced binary tree with \(q\) leaves. Figure 2 illustrates the construction of degree 4, \(Q^4\).

For every integer \(q\) which is a power of two, by definition of \(S^q\) in (3.2) of Definition 3.1, \(S^q = S^2 \times \cdots \times S^q_{q/2}\). Hence, by claim 1 we can write,

\[
S^q = S^q_1 \times \cdots \times S^q_{q/2} = \left( S^2_1 \times \cdots \times S^2_{q-2} \times I_m \right) \left( I_{m^{q/2-2}} \times S^q_{q/2} \right).
\]

By multiple applications of Claim 1 we have the following claim,

**Claim 2.** For every power of two integer \(q\) and any positive integer \(m\), if \(S^q\) is defined as in (3.2) of Definition 3.1 then \(S^q = M_{q/2} M_{q/2-1} \cdots M_1\), where \(M_j = I_{m^{q/2-j}} \times S^q_{q/2-j+1} \times I_{m_j^{-1}}\) for every \(j \in [q/2]\).

---

**Embedding \(\mathbb{R}^d\):** So far we have constructed a sketch \(Q^q\) for sketching tensor product of vectors in \(\mathbb{R}^m\). However, in general the data points can be in a space \(\mathbb{R}^d\) of arbitrary dimension. A natural idea is to reduce the dimension of the vectors by a mapping from \(\mathbb{R}^d\) to \(\mathbb{R}^m\) and then apply \(Q^q\) on the tensor product of reduced data points. The dimensionality reduction defines a linear mapping from \(\mathbb{R}^d\) to \(\mathbb{R}^{m^q}\) which can be represented by a matrix. We denote the dimensionality reduction matrix by \(T^q \in \mathbb{R}^{m^q \times d^q}\) formally defined as follows.

**Definition 3.2.** Let \(m, d\) be positive integers and let \(T_{\text{base}} : \mathbb{R}^d \rightarrow \mathbb{R}^m\) be a linear map that specifies some base sketch. Then for any integer power of two \(q\) we define \(T^q\) to be the linear map specified as follows,

\[
T^q = T_1 \times T_2 \times \cdots \times T_q,
\]

where the matrices \(T_1, \ldots, T_q\) are drawn independently from \(T_{\text{base}}\).

**Discussion:** Similar to Claim 2, the transform \(T^q\) can be expressed as the following product of \(q\) matrices,

\[
T^q = M_q \cdots M_1,
\]

where \(M_j = I_{d^{q-j}} \times T_{q-j+1} \times I_{m_j^{-1}}\) for every \(j \in [q]\).

Now we define the final sketch \(\Pi^q : \mathbb{R}^d \rightarrow \mathbb{R}^m\) for arbitrary \(d\) as the composition of \(Q^q \cdot T^q\). Moreover, to extend the definition to arbitrary \(q\) which is not necessarily a power of two we tensor the input vector with a standard basis vector a number of times to make the input size compatible with the sketch matrices. The sketch \(\Pi^q\) is formally defined below.

**Definition 3.3. (Sketch \(\Pi^q\))** Let \(m, d\) be positive integers and let \(S_{\text{base}} : \mathbb{R}^{m^2} \rightarrow \mathbb{R}^m\) and \(T_{\text{base}} : \mathbb{R}^d \rightarrow \mathbb{R}^m\) be linear maps that specify some base sketches. Then,
for any integer \( p \geq 2 \) we define \( \Pi^p : \mathbb{R}^{dp} \rightarrow \mathbb{R}^m \) to be the linear map specified as follows:

1. If \( p \) is a power of two, then \( \Pi^p \) is defined as
   \[ \Pi^p = Q^p \cdot T^p, \]
   where \( Q^p \in \mathbb{R}^{m \times m^p} \) and \( T^p \in \mathbb{R}^{m^p \times dp} \) are sketches as in Definitions 3.1 and 3.2 respectively.

2. If \( p \) is not a power of two, then let \( q = 2^\lfloor \log_2 p \rfloor \) be the smallest power of two integer that is greater than \( p \) and we define \( \Pi^p \) as
   \[ \Pi^p(v) = \Pi^q \left(v \otimes e_1^{\otimes (q-p)}\right), \]
   for every \( v \in \mathbb{R}^{dp} \), where \( e_1 \in \mathbb{R}^d \) is the standard basis column vector with a 1 in the first coordinate and zeros elsewhere, and \( \Pi^q \) is defined as in the first part of this definition.

Algorithm 1 sketches \( x \otimes p \) for any integer \( p \) and any input vector \( x \in \mathbb{R}^d \) using the sketch \( \Pi^p \) as in Definition 3.3, i.e., computes \( \Pi^p(x \otimes p) \).

**Algorithm 1 Sketch for the Tensor \( x \otimes p \)**

**input:** vector \( x \in \mathbb{R}^d \), dimension \( d \), degree \( p \), number of buckets \( m \), base sketches \( S_{base} \in \mathbb{R}^{m \times m^2} \) and \( T_{base} \in \mathbb{R}^{m \times d} \)

**output:** sketched vector \( z \in \mathbb{R}^m 

```markdown
1. Let \( q = 2^\lfloor \log_2 p \rfloor \)
2. Let \( T_1, \cdots, T_q \) be independent instances of the base sketch \( T_{base} : \mathbb{R}^d \rightarrow \mathbb{R}^m \)
3. For every \( j \in \{1, 2, \cdots, p\} \), let \( Y_j^0 = T_j \cdot x \)
4. For every \( j \in \{p+1, \cdots, q\} \), let \( Y_j^0 = T_j \cdot e_1 \), where \( e_1 \) is the standard basis vector in \( \mathbb{R}^d \) with value 1 in the first coordinate and zeros elsewhere
5. For \( l = 1 \) to \( \log_2 q \)
6. Let \( S_{q/2}^{l/2-1}, \cdots, S_{q/2}^{l/2-1} \) be independent instances of the base sketch \( S_{base} : \mathbb{R}^n \rightarrow \mathbb{R}^m \)
7. For every \( j \in \{1, \cdots, q/2^l\} \) let \( Y_j^l = S_{j}^{l/2^l} \left(Y_{2j-1}^{l-1} \otimes Y_{2j}^{l-1}\right) \)
8. end for
9. return \( z = Y_1^{\log_2 q} \)
```

Next lemma shows the correctness of Algorithm 1.

**Lemma 3.1.** For any positive integers \( d, m, \) and \( p \), any distribution on matrices \( S_{base} : \mathbb{R}^{m^2} \rightarrow \mathbb{R}^m \) and \( T_{base} : \mathbb{R}^d \rightarrow \mathbb{R}^m \) which specify some base sketches, any vector \( x \in \mathbb{R}^d \), Algorithm 1 computes \( \Pi^p(x \otimes p) \) as in Definition 3.3.

The proof is available in the full version of this work.

**Choices of the Base Sketches \( S_{base} \) and \( T_{base} \):**

We present formal definitions for various base sketches \( S_{base}, T_{base} \) that will be used in our sketch construction. We start by briefly recalling the CountSketch \([CCFC02]\).

**Definition 3.4.** (CountSketch transform) Let \( h : [d] \rightarrow [m] \) be a 3-wise independent hash function and also let \( \sigma : [d] \rightarrow \{-1, +1\} \) be a 4-wise independent random sign function. Then, the CountSketch transform, \( S : \mathbb{R}^d \rightarrow \mathbb{R}^m \), is defined as follows; for every \( i \in [d] \) and every \( r \in [m] \), \( S_r = \sigma(i) \cdot 1[h(i) = r] \).

Another base sketch that we consider is the TensorSketch of degree two \([Pag13]\) defined as follows.

**Definition 3.5.** *(Degree two TensorSketch)* Let \( h_1, h_2 : [d] \rightarrow [m] \) be 3-wise independent hash functions and also let \( \sigma_1, \sigma_2 : [d] \rightarrow \{-1, +1\} \) be 4-wise independent random sign functions. Then, the degree two TensorSketch transform, \( S : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^m \), is defined as follows; for every \( i, j \in [d] \) and every \( r \in [m] \), \( S_{r, (i,j)} = \sigma_1(i) \cdot \sigma_2(j) \cdot 1[h_1(i) + h_2(j) = r \mod m] \).

**Remark:** \( S(x \otimes 2) \) can be computed in \( O(m \log m + m \log 2(x)) \) time using the Fast Fourier Transform.

Now let us briefly recall the Subsampled Randomized Hadamard Transform (SRHT) \([AC06]\).

**Definition 3.6.** *(SRHT)* Let \( D \) be a \( d \times d \) diagonal matrix with independent Rademacher random variables along the diagonal. Also, let \( P \in \{0, 1\}^{m \times d} \) be a random sampling matrix in which each row contains a 1 at a uniformly distributed coordinate and zeros elsewhere, and let \( H \) be a \( d \times d \) Hadamard matrix. Then, the SRHT, \( S \in \mathbb{R}^{m \times d} \), is \( S = \frac{1}{\sqrt{m}} P D H \).

We now define a variant of the SRHT which is very efficient for sketching \( x \otimes 2 \) which we call the TensorSRHT.

**Definition 3.7.** *(TensorSRHT)* Let \( D_1 \) and \( D_2 \) be two independent \( d \times d \) diagonal matrices, each with diagonal entries given by iid Rademacher variables. Also let \( P \in \{0, 1\}^{m \times d} \) be a random sampling matrix in which each row contains exactly one uniformly distributed nonzero element which has value one, and let \( H \) be a \( d \times d \) Hadamard matrix. Then, the TensorSRHT is defined to be \( S : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^m \) given by \( S = \frac{1}{\sqrt{m}} P \cdot (H D_1 \times H D_2) \).

**Remark:** \( S(x \otimes 2) \) can be computed in time \( O(d \log d + m) \) using the FFT algorithm.

A sketch which is particularly efficient for sketching sparse vectors with high probability is OSNAP \([NN13]\).
DEFINITION 3.8. (OSNAP Transform) For every sparsity parameter \( s \), target dimension \( m \), and positive integer \( d \), the OSNAP transform with sparsity parameter \( s \) is defined as,

\[
S_{r,j} = \sqrt{\frac{t}{s}} \cdot \delta_{r,j} \cdot \sigma_{r,j},
\]

for all \( r \in [m] \) and all \( j \in [d] \), where \( \sigma_{r,j} \in \{-1, +1\} \) are independent and uniform Rademacher random variables and \( \delta_{r,j} \) are Bernoulli random variables satisfying,

1. For every \( i \in [d] \), \( \sum_{r \in [m]} \delta_{r,i} = s \). That is, each column of \( S \) has exactly \( s \) non-zero entries.

2. For all \( r \in [m] \) and all \( i \in [d] \), \( E[\delta_{r,i}] = s/m \).

3. The \( \delta_{r,i} \)'s are negatively correlated: \( \forall T \subset [m] \times [d] \), \( E[\prod_{(r,i) \in T} \delta_{r,i}] \leq \prod_{(r,i) \in T} E[\delta_{r,i}] = \left( \frac{s}{m} \right)^{|T|} \).

4 Linear Dependence on the Tensoring Degree

There are various desirable properties that we would like to have a linear sketch to satisfy. One such property which is central to our main results is the JL Moment Property. In this section we prove Theorem 1.1 and Theorem 1.2 by propagating the JL Moment Property through our recursive construction from Section 3. The JL Moment Property captures a bound on the moments of the difference between the Euclidean norm of a vector and its Euclidean norm after applying the sketch on it. The JL Moment Property proves to be a powerful property for a sketch and we will show that it implies the Oblivious Subspace Embedding as well as the Approximate Matrix Product property for linear sketches.

In section 4.1 we choose \( S_{\text{base}} \) and \( T_{\text{base}} \) to be TensorSketch and CountSketch respectively. Then we propagate the second JL Moment through the sketch construction \( \Pi^p \) and thereby prove Theorem 1.1. In section 4.2 we propagate the higher JL Moments through our recursive construction \( \Pi^p \) as per Definition 3.3 with TensorSRHT at the internal nodes (\( S_{\text{base}} \)) and OSNAP at the leaves (\( T_{\text{base}} \)), thereby proving Theorem 1.2.

To make the notation less heavy we will use \( \|X\|_{L^t} \) for the \( t \)th moment of a random variable \( X \). This is formally defined below.

DEFINITION 4.1. For every integer \( t \geq 1 \) and any random variable \( X \in \mathbb{R} \), we write

\[
\|X\|_{L^t} = \left( E \left[ |X|^t \right] \right)^{1/t}.
\]

Note that \( \|X + Y\|_{L^t} \leq \|X\|_{L^t} + \|Y\|_{L^t} \) for any random variables \( X, Y \) by the Minkowski Inequality.

We now formally define the JL Moment Property of sketches.

DEFINITION 4.2. (JL Moment Property) For every positive integer \( t \) and every \( \delta, \varepsilon \geq 0 \), we say a distribution over random matrices \( S \in \mathbb{R}^{m \times d} \) has the \((\varepsilon, \delta, t)\)-JL-moment property, when

\[
\||Sx\|^2_t - 1||_{L^t} \leq \varepsilon \delta^{1/t} \quad \text{and} \quad E[||Sx||^2_t] = 1
\]

for all \( x \in \mathbb{R}^d \) such that \( \|x\| = 1 \).

The JL Moment Property directly implies the following moment bound for the inner product of two vectors:

LEMMA 4.1. (Two Vector JL Moment Property) For every \( x, y \in \mathbb{R}^d \), if \( S \) has the \((\varepsilon, \delta, t)\)-JL Moment Property, then

\[
\left\| (Sx) \cdot (Sy) - x^T y \right\|_{L^t} \leq \varepsilon \delta^{1/t} \|x\|_2 \|y\|_2.
\]

The proof is available in the long version of this paper.

We will also need the Strong JL Moment Property, which is a sub-Gaussian bound on the difference between the Euclidean norm of a vector and its Euclidean norm after applying the sketch on it.

DEFINITION 4.3. (Strong JL Moment Property) For every \( \varepsilon, \delta > 0 \) we say a distribution over random matrices \( M \in \mathbb{R}^{m \times d} \) has the Strong \((\varepsilon, \delta)\)-JL Moment Property when

\[
\left\| \|Mx\|^2_t - 1 \right\|_{L^t} \leq \sqrt{\frac{t}{\varepsilon}} \frac{1}{\log(1/\delta)} \quad \text{and} \quad E[\|Mx\|^2_t] = 1,
\]

for all \( x \in \mathbb{R}^d \), \( \|x\| = 1 \) and every integer \( t \leq \log(1/\delta) \).

REMARK 4.1. It should be noted that if a matrix \( M \in \mathbb{R}^{m \times d} \) has the Strong \((\varepsilon, \delta)\)-JL Moment Property then it has the \((\varepsilon, \delta, \log(1/\delta))\)-JL Moment Property, since

\[
\left\| \|Mx\|^2_t - 1 \right\|_{L^t} \leq \sqrt{\frac{1}{\varepsilon \delta \log(1/\delta)}} = \frac{\varepsilon}{\delta} = \varepsilon \delta^{\log(1/\delta)}.
\]

The following lemma shows that if we want to prove that \( \Pi^p \) is an Oblivious Subspace Embedding and that \( \Pi^p \) has the Approximate Matrix Multiplication Property, then it suffices to prove that \( \Pi^p \) has the JL Moment Property. This reduction will be the main component of the proofs of Theorem 1.1 and Theorem 1.2.

LEMMA 4.2. For any \( \varepsilon, \delta \in [0, 1] \), \( t \geq 1 \), if \( M \in \mathbb{R}^{m \times d} \) is a random matrix with \((\varepsilon, \delta, t)\)-JL Moment Property then \( M \) has the \((\varepsilon, \delta)\)-Approximate Matrix Multiplication Property.

Furthermore, for any \( \mu > 0 \), if \( M \in \mathbb{R}^{m \times d} \) is a random matrix with \((\varepsilon/\mu, \delta, t)\)-JL Moment Property then for every positive integer \( n \in \mathbb{Z} \), \( M \) is a \((\varepsilon, \delta, \mu, d, n)\)-OSE.
Our next important observation is that $\Pi^q$ can be written as the product of $2q - 1$ independent random matrices, which all have a special structure which makes them easy to analyse.

**Lemma 4.3.** For any integer $q$ which is a power of two, $\Pi^q : \mathbb{R}^{m^q} \to \mathbb{R}^m$ be defined as in Definition 4.3 for some base sketches $S_{base} : \mathbb{R}^{m^q} \to \mathbb{R}^m$ and $T_{base} : \mathbb{R}^d \to \mathbb{R}^{m^q}$. Then there exist matrices $(M^{(i)})_{i \in [q-1]}, (M^{(j)})_{j \in [q]}$ and integers $(k_i)_{i \in [q-1]}, (k_j)_{j \in [q]}, (l_i)_{i \in [q]}, (l_j)_{j \in [q]}$, such that,

$$\Pi^q = M^{(q-1)} \cdot \ldots \cdot M^{(1)} \cdot M^{(q)} \cdot \ldots \cdot M^{(1)},$$

and $M^{(i)} = I_{k_i} \times S_{base}^{(i)} \times I_{k_i'}$, $M^{(j)} = I_{l_j} \times T_{base}^{(j)} \times I_{l_j'}$, where $S_{base}^{(i)}$ and $T_{base}^{(j)}$ are independent instances of $S_{base}$ and $T_{base}$, for every $i \in [q-1], j \in [q]$.

The following simple fact shows that $I_k \times M \times I_k'$ inherits the JL properties of $M$.

**Lemma 4.4.** Let $t \in \mathbb{N}$ and $\alpha \geq 0$. If $P \in \mathbb{R}^{m_1 \times d_1}$ and $Q \in \mathbb{R}^{m_2 \times d_2}$ are two random matrices (not necessarily independent), such that,

$$\|P \times x\|_2^2 - \|x\|_2^2 \leq \alpha \|x\|_2^2 \quad \text{and} \quad \mathbb{E}[\|P \times x\|_2^2] = \|x\|_2^2,$$

$$\|Q \times y\|_2^2 - \|y\|_2^2 \leq \alpha \|y\|_2^2 \quad \text{and} \quad \mathbb{E}[\|Q \times y\|_2^2] = \|y\|_2^2,$$

for any vectors $x \in \mathbb{R}^{d_1}$ and $y \in \mathbb{R}^{d_2}$, then

$$\|((P \oplus Q)z)\|_2^2 - \|z\|_2^2 \leq \alpha \|z\|_2^2$$

and

$$\mathbb{E}[\|((P \oplus Q)z)\|_2^2] = \|z\|_2^2,$$

for any vector $z \in \mathbb{R}^{d_1 + d_2}$.

This lemma is proved in the long version of this paper. An easy consequence of this lemma is that for any matrix, $S$, with the $(\varepsilon, \delta, t)$-JL Moment Property, $I_k \times S$ has the $(\varepsilon, \delta, t)$-JL Moment Property. This follows simply from $I_k \times S = S \oplus S \oplus \ldots \oplus S$. Similarly, $S \times I_k$ has the $(\varepsilon, \delta, t)$-JL Moment Property, since $S \times I_k$ is just a reordering of the rows of $I_k \times S$, which trivially does not affect the JL Moment Property. The same arguments show that if $S$ has the Strong $(\varepsilon, \delta)$-JL Moment Property then $I_k \times S$ and $S \times I_k$ has the Strong $(\varepsilon, \delta)$-JL Moment Property. So we conclude the following

**Lemma 4.5.** If the matrix $S$ has the $(\varepsilon, \delta, t)$-JL Moment Property, then for any positive integers $k, k'$, the matrix $M = I_k \times S \times I_{k'}$ has the $(\varepsilon, \delta, t)$-JL Moment Property. Similarly, if the matrix $S$ has the Strong $(\varepsilon, \delta)$-JL Moment Property, then for any positive integers $k, k'$, the matrix $M = I_k \times S \times I_{k'}$ has the Strong $(\varepsilon, \delta)$-JL Moment Property.

Now if we can prove that the product of matrices with the JL Moment Property has the JL Moment Property, then Lemmas 4.4 and 4.3 would imply that $\Pi^q$ has the JL Moment Property, which again implies that $\Pi^q$ is an Oblivious Subspace Embedding and has the Approximate Matrix Multiplication Property, by Lemma 4.2. This is exactly what we will do: In Section 4.1 we prove that the product of $k$ independent matrices with the $(\varepsilon, \delta, 2)$-JL Moment Property results in a matrix with the $(\varepsilon, \delta, 2)$-JL Moment Property, which will give us the proof of Theorem 4.1 and in Section 4.2 we prove that the product of $k$ independent matrices with the Strong $(\varepsilon, \delta)$-JL Moment Property results in a matrix with the Strong $(\varepsilon, \delta)$-JL Moment Property, which will give us the proof of Theorem 4.2.

### 4.1 Second Moment of $\Pi^q$ (analysis for $T_{base} : \text{CountSketch}$ and $S_{base} : \text{TensorSketch}$)

In this section we prove Theorem 1.1 by instantiating our recursive construction from Section 3 with $\text{CountSketch}$ at the leaves and $\text{TensorSketch}$ at the internal nodes of the tree. The proof proceeds by showing the second moment property – i.e., $(\varepsilon, \delta, 2)$-JL Moment Property, for our recursive construction. We prove that our sketch $\Pi^q$ satisfies the $(\varepsilon, \delta, 2)$-JL Moment Property as per Definition 4.2 as long as the base sketches $S_{base}, T_{base}$ are chosen from a distribution which satisfies the second moment property. We show that this is the case for $\text{CountSketch}$ and $\text{TensorSketch}$.

Lemma 4.5 together with Lemma 4.3 show that if the base sketches $S_{base}, T_{base}$ have the JL Moment Property then $\Pi^q$ is the product of $2q - 1$ independent random matrices with the JL Moment Property. Therefore, understanding how matrices with the JL Moment Property compose is crucial. The following lemma shows that composing independent random matrices which have the JL Moment Property results in a matrix which has the JL Moment Property with a small loss in the parameters.

**Lemma 4.6.** (Composition Lemma, 2nd Moment) For any $\varepsilon, \delta \geq 0$ and any integer $k$ if $M^{(1)} \in \mathbb{R}^{d_1 \times d_1}, \ldots, M^{(k)} \in \mathbb{R}^{d_k \times d_k}$ are independent random matrices with the $(\frac{\varepsilon}{\sqrt{2k}}, \delta, 2)$-JL-moment property then the product matrix $M = M^{(k)} \cdot \ldots \cdot M^{(1)}$ satisfies the $(\varepsilon, \delta, 2)$-JL-moment property.

This lemma is proved by application of the law of total variance and using the definition of JL-moment property in the long version of this paper. Equipped with the composition lemma for the second moment, we now establish the second moment property for our recursive
TensorSRHT 

construction of Section 3 with prove Theorem 1.2 by instantiating our recursive con-

OSNAP

Π

4.2 Higher Moments of \( \Pi' \)

Theorem 1.1 Recall that \( k(x, y) = \langle x, y \rangle^q \) is the polynomial kernel of degree \( q \). One can see that \( k(x, y) = \langle x \otimes q, y \otimes q \rangle \).

Let \( x_1, x_2, \ldots, x_n \in \mathbb{R}^m \) be an arbitrary dataset of \( n \) points in \( \mathbb{R}^m \). We represent the data points by matrix \( X \in \mathbb{R}^{m \times n} \) whose \( i \)th column is the vector \( x_i \). Let \( A \in \mathbb{R}^{m \times r} \) be the matrix whose \( i \)th column is \( x_i \otimes q \) for every \( i \in [n] \). For any regularization parameter \( \lambda > 0 \), the statistical dimension of \( A^T A \) is defined as \( s_A := \text{tr} \left( (A^T A)(A^T A + \lambda I)^{-1} \right) \).

Proof of Theorem 4.1 Let \( \Pi' \in \mathbb{R}^{m \times d^p} \) be the sketch defined in Definition 3.3, where the base distributions \( S_{\text{base}} \in \mathbb{R}^{m \times m^2} \) and \( T_{\text{base}} \in \mathbb{R}^{m \times d} \) are respectively the standard TensorSketch of degree two and standard CountSketch. It is shown in [ANW14] and [CW17] that for these choices of base sketches, \( S_{\text{base}} \) and \( T_{\text{base}} \) both satisfy the \( (\sqrt{\frac{\epsilon}{4p+2}}, \delta, 2) \)-JL-moment property as long as \( m = \Omega(\frac{r}{\epsilon^2}) \) (see Definition 4.2). Thus using Corollary 4.2 we conclude that \( \Pi' \) has the \( (\epsilon, \delta, 2) \)-JL Moment Property. Thus, the correctness follows from Lemma 4.2.

Runtime of Algorithm 1 when the base sketch \( S_{\text{base}} \) is TensorSketch of degree two and \( T_{\text{base}} \) is CountSketch: We compute the time of running Algorithm 1 on a vector \( x \). Computing \( Y^0_j \) for each \( j \) in lines 3 and 4 of algorithm requires applying a CountSketch on either \( x \) or \( e_1 \) which takes time \( O(\text{nnz}(x)) \). Therefore computing all \( Y^0_j \)'s takes time \( O(q \cdot \text{nnz}(x)) \).

Computing each of \( Y^l_j \)'s for \( l \geq 1 \) in line 7 of Algorithm 1 amounts to applying a degree two TensorSketch of input dimension \( m^2 \) and target dimension of \( m \) on \( Y^{l-1}_{2j-1} \otimes Y^{l-1}_{2j-1} \). This takes time \( O(m \log m) \). Therefore computing \( Y^l_j \) for all \( l, j \geq 1 \) takes time \( O(q \cdot m \log m) \). Note that \( q \leq 2p \) and hence the total running time of Algorithm 1 on one vector \( x \) is \( O(p \cdot m \log_2 m + p \cdot \text{nnz}(w)) \).

Sketching \( n \) columns of a matrix \( X \in \mathbb{R}^{d \times n} \) takes time \( O(p \cdot m \log_2 m + \text{nnz}(X)) \). \( \square \)

4.2 Higher Moments of \( \Pi' \) (analysis for \( T_{\text{base}} : \text{OSNAP} \) and \( S_{\text{base}} : \text{TensorSRHT} \)) In this section we prove Theorem 4.2 by instantiating our recursive construction of Section 3 with OSNAP at the leaves and TensorSRHT at the internal nodes.

The proof proceeds by showing the Strong JL Moment Property for our sketch \( \Pi' \). If a sketch satisfies the Strong JL Moment Property then it straightforwardly is an OSE and has the approximate matrix product property.

We will need Khintchine’s inequality.

Lemma 4.7. (Khintchine’s Inequality \([HM07]\]) Let \( t \) be a positive integer, \( x \in \mathbb{R}^d \), and \( (\sigma_i)_{i \in [d]} \) be independent Rademacher \( \pm 1 \) random variables. Then

\[
\left\| \sum_{i=1}^{d} \sigma_i x_i \right\|_{L^t} \leq C_t \|x\|_2,
\]

where \( C_t \leq \sqrt{2^{1/t} \left( \frac{(1+t)}{2} \right)^{1/t}} \leq \sqrt{t} \) for all \( t \geq 1 \).

One may replace \( (\sigma_i) \) with an arbitrary independent sequence of random variables \( (\xi_i) \) with \( E[\xi_i] = 0 \) and \( \|\xi_i\|_{L^t} \leq \sqrt{t} \) for any \( 1 \leq t \leq t \), and the lemma still holds up to a universal constant factor on the r.h.s.

First we note that the OSNAP transform satisfies the strong JL moment property (proof in long version).

Lemma 4.8. There exists a universal constant \( L \), such that, the following holds. Let \( M \in \mathbb{R}^{m \times d} \) be an OSNAP transform with sparsity parameter \( s \). Let \( x \in \mathbb{R}^d \) be any vector with \( \|x\|_2 = 1 \) and \( t \geq 1 \), then

\[
\left\| \|Mx\|_2^t - 1 \right\|_{L^t} \leq L \left( \sqrt{\frac{t}{m}} + \frac{t}{s} \right).
\]

Setting \( m = \Omega(\epsilon^{-2} \log(1/\delta)) \) and \( s = \Omega(\epsilon^{-1} \log(1/\delta)) \) then \( M \) has the Strong \( (\epsilon, \delta) \)-JL Moment Property (Definition 4.3).

We continue by proving that SRHT and TensorSRHT sketches satisfy the strong JL moment property. We will do this by proving that a more general class of matrices satisfies the strong JL moment property. More precisely, let \( k \in \mathbb{Z}_{>0} \) be a positive integer and \( (D(i))_{i \in [k]} \in \prod_{i \in [k]} \mathbb{R}^{d \times d} \) be independent matrices, each with diagonal entries given by independent Rademacher variables. Let \( d = \prod_{i \in [k]} d_i \), and \( P \in \{0, 1\}^{m \times d} \) be a random sampling matrix in which each row contains exactly one uniformly distributed nonzero element which has value one. Then we will prove that the matrix \( M = \frac{1}{\sqrt{m}} PH(D_1 \times \ldots \times D_k) \) satisfies the strong JL moment property, where \( H \) is a \( d \times d \)
Hadamard matrix. If \( k = 1 \) then \( M \) is just a SRHT, and if \( k = 2 \) then \( M \) is a TensorSRHT.

In order to prove this result we need a lemma which can be seen as a version of Khintchine’s inequality for higher order chaos.

**Lemma 4.9.** Let \( t \geq 1, k \in \mathbb{Z}_{>0}, \) and \((\sigma^{(i)})_{i \in [k]} \in \prod_{i \in [k]} \mathbb{R}^{d_i}\) be independent vectors each satisfying the Khintchine inequality \( \| (\sigma^{(i)}, x) \|_{L_t} \leq C_t \| x \|_2 \) for \( t \geq 1 \) and any vector \( x \in \mathbb{R}^{d_i}. \) Let \((a_{i_1, \ldots, i_k})_{i_1 \in [d_1], \ldots, i_k \in [d_k]} \) be a tensor in \( \mathbb{R}^{d_1 \times \cdots \times d_k}, \) then

\[
\left\| \sum_{i_1 \in [d_1], \ldots, i_k \in [d_k]} \left( \prod_{j \in [k]} \sigma^{(j)}_{i_j} \right) a_{i_1, \ldots, i_k} \right\|_{L_t} \leq C_t^k \left( \sum_{i_1 \in [d_1], \ldots, i_k \in [d_k]} a_{i_1, \ldots, i_k}^2 \right)^{1/2},
\]

for \( t \geq 1. \) Or, considering \( a \in \mathbb{R}^{d_1 \times \cdots \times d_k} \) a vector, then simply \( \| (\sigma^{(1)} \otimes \cdots \otimes \sigma^{(k)}, a) \|_{L_t} \leq C_t^k \| a \|_2, \) for \( t \geq 1. \)

This is related to Latala’s estimate for Gaussian chaoses [Lat06], but more simple in the case where \( a \) is not assumed to have special structure. Note that this implies the classical bound on the fourth moment of products of \( 4 \)-wise independent hash functions [BCL+10] [M08] [PT12], since \( C_t = 3^{1/4} \) for Rademachers we have\( \mathbb{E}[\sigma^{(1)} \otimes \cdots \otimes \sigma^{(k)}, x]^4] \leq 3^k \| x \|_2^4 \) for four-independent \((\sigma^{(i)})_{i \in [k]} \). The complete proof of this paper is present in the full version of this paper.

The next lemma we will be using is a type of Rosenthal inequality, but which mixes large and small moments in a careful way. It bears similarity to the one sided bound in [BLM13] (Theorem 15.10) derived from the Efron Stein inequality, and the literature has many similar bounds, but we still include a proof here based on first principles.

**Lemma 4.10.** There exists a universal constant \( L, \) such that, for \( t \geq 1 \) and \( X_1, \ldots, X_k \) be independent non-negative random variables with \( t \)-moment, then

\[
\left\| \sum_{i \in [k]} (X_i - \mathbb{E}[X_i]) \right\|_{L_t} \leq L \left( \sqrt{t} \left\| \max_{i \in [k]} X_i \right\|_{L_t}^{1/2} \sqrt{\sum_{i \in [k]} \mathbb{E}[X_i]} + t \right) \left\| \max_{i \in [k]} X_i \right\|_{L_t}^{1/2}.
\]

We can now prove that SRHT and TensorSRHT has the Strong JL Moment Property.

**Lemma 4.11.** There exists a universal constant \( L, \) such that, the following holds. Let \( k \in \mathbb{Z}_{>0}, \) and \((D^{(i)})_{i \in [k]} \in \prod_{i \in [k]} \mathbb{R}^{d_i \times d_i} \) be independent diagonal matrices with independent Rademacher variables. Define \( d = \prod_{i \in [k]} d_i \) and \( D = D_1 \times D_2 \times \cdots \times D_k \in \mathbb{R}^{d \times d}. \) Let \( F \in \mathbb{R}^{m \times d} \) be an independent sampling matrix which samples exactly one coordinate per row, and define \( M = PHD \) where \( H \) is a \( d \times d \) Hadamard matrix. Let \( x \in \mathbb{R}^d \) be any vector with \( \| x \|_2 = 1 \) and \( t \geq 1, \) then

\[
\frac{1}{m} \| PHD x \|_2^2 - 1 \|_{L_t} \leq L \left( \frac{\sqrt{tr_d k}}{m} + \frac{ts_k}{m} \right),
\]

where \( r = \max\{t, \log m\}. \) There exists a universal constant \( L', \) such that, setting \( m = \Omega(\varepsilon^{-2}\log(1/\delta)(L'/\log(1/\varepsilon))^{k}) \), we get that \( \frac{1}{\sqrt{m}} PHD \) has Strong (\( \varepsilon, \delta \))-JL Moment Property.

Note that setting \( k = 1, \) this matches the Fast Johnson Lindenstrauss analysis in [CNW16].

Now we have proved that the Strong JL Moment Property is satisfied by the SRHT, the TensorSRHT as well as OSNAP transform, but we still need to prove the usefulness of the property. Our next result remedies this and show that the Strong JL Moment Property is preserved under multiplication. We will use the following decoupling lemma which first appeared in [Hit94], but the following is roughly taken from [DIPG12] (Theorem 7.3.1), which we also recommend for readers interested in more general versions.

**Lemma 4.12.** (General Decoupling) There exists an universal constant \( C_0, \) such that, given any two sequences \((X_i)_{i \in [n]} \) and \((Y_i)_{i \in [n]} \) of random variables where \((X_i)_{i \in [n]} \) is adapted to an increasing filtration \((\mathcal{F}_i)_{i \in [n]} \), satisfying

1. \( \Pr[Y_i > t \mid \mathcal{F}_{i-1}] = \Pr[X_i > t \mid \mathcal{F}_{i-1}] \) for every \( t \in \mathbb{R} \) and for every \( i \in [n]. \)

2. The sequence \((Y_i)_{i \in [n]} \) is conditionally independent given \( \mathcal{F}_n. \)

3. \( \Pr[Y_i > t \mid \mathcal{F}_{i-1}] = \Pr[Y_i > t \mid \mathcal{F}_n] \) for every \( t \in \mathbb{R} \) and for every \( i \in [n]. \)

Then for all \( t \geq 1, \)

\[
\left\| \sum_{i \in [n]} X_i \right\|_{L_t} \leq C_0 \left\| \sum_{i \in [n]} Y_i \right\|_{L_t}.
\]

We are now ready to state and prove the main lemma of this section.

**Lemma 4.13.** There exists a universal constant \( L, \) such that, for any constants \( \varepsilon, \delta \in [0, 1] \) and positive integer \( k \in \mathbb{Z}_{>0}. \) If \( M^{(k)} \in \mathbb{R}^{d_1 \times d_1, \ldots, M^{(k)} \in \mathbb{R}^{d_{k+1} \times d_k} \) are independent random matrices with the
Khintchine’s inequality (Lemma 4.7), we get that

\[ \text{Proof.} \]

Let \( x \in \mathbb{R}^d \) be an arbitrary, fixed unit vector, and fix \( 1 < t \leq \log(1/\delta) \). We define \( X_i = \|M^{(i)} \ldots M^{(1)} x\|_2^2 \) and \( Y_i = X_i - X_{i-1} \) for every \( i \in [k] \). By telescoping we then have that \( X_1 - 1 = \sum_{j \in [i]} Y_i \). We let \( \mathcal{T}^{(i)} \) be independent copies of \( (M^{(i)} )_{i \in [k]} \) and define

\[ Z_i = \| \mathcal{T}^{(i)} \cdot M^{(i-1)} \ldots M^{(1)} x \|_2^2 - \| M^{(i-1)} \ldots M^{(1)} x \|_2^2, \]

for every \( i \in [k] \). Define the filtration \( \mathcal{F}_i = (M^{(j)})_{j \in [i]} \). We get the following three properties:

1. \( \Pr[Z_i > t \mid \mathcal{F}_{i-1}] = \text{Pr}[Y_i > t \mid \mathcal{F}_{i-1}] \) for every \( t \in \mathbb{R} \) and every \( i \in [k] \).
2. The sequence \( (Z_i)_{i \in [k]} \) is conditionally independent given \( \mathcal{F}_k \).
3. \( \Pr[Z_i > t \mid \mathcal{F}_{i-1}] = \text{Pr}[Z_i > t \mid \mathcal{F}_k] \) for every \( t \in \mathbb{R} \) and for every \( i \in [k] \).

This means we can use Lemma 4.12 to get

\[ \left\| \sum_{j \in [i]} Y_j \right\|_{L_t'} \leq C_0 \left\| \sum_{j \in [i]} Z_j \right\|_{L_t'}, \]

for every \( i \in [k] \).

We will prove by induction on \( i \in [k] \) that

\[ X_i - 1 \leq \frac{\varepsilon}{c} \sqrt{\frac{t}{\log(1/\delta)}} \leq 1. \]

For \( i = 1 \) we use that \( M^{(1)} \) has the Strong \((\varepsilon/(L\sqrt{k}), \delta)\)-JL Moment Property and get that

\[ \left\| M^{(1)} x \right\|_2^2 - 1 \right\|_{L_t'} \leq \frac{\varepsilon}{c L \sqrt{k}} \sqrt{\frac{t}{\log(1/\delta)}} \leq \frac{\varepsilon}{c} \sqrt{\frac{t}{\log(1/\delta)}}. \]

Now assume that (4.6) is true for \( i - 1 \). Using (4.5) we get that \( \left\| X_{i-1} \right\|_{L_t'} = \left\| \sum_{j \in [i]} Y_j \right\|_{L_t'} \leq C_0 \left\| \sum_{j \in [i]} Z_j \right\|_{L_t'} \). By using that \( (T^{(j)})_{j \in [i]} \) has the Strong \((\varepsilon/(L\sqrt{k}), \delta)\)-JL Moment Property together with Khintchine’s inequality (Lemma 4.7), we get that

\[ \left\| \sum_{j \in [i]} Z_j \right\|_{L_t'} \leq C_1 \varepsilon \sqrt{\frac{t}{\log(1/\delta)}} \frac{1}{L \sqrt{k}} \left\| \sum_{j \in [i]} X_j \right\|_{L_t'}, \]

where the last inequality follows from the triangle inequality. Using the triangle inequality and (4.6) we get that

\[ \| X_j \|_{L_t'} \leq 1 + \| X_j - 1 \|_{L_t'} \leq 2, \]

for every \( j \in [i] \). Setting \( L = 2C_0C_1 \) we get that

\[ \left\| \sum_{j \in [i]} Y_j \right\|_{L_t'} \leq \frac{\varepsilon}{c} \sqrt{\frac{t}{\log(1/\delta)}} \cdot \frac{C_0C_1}{L \sqrt{k}} \sqrt{\sum_{j \in [i]} \| X_j \|_{L_t'}^2} \]

\[ \leq \frac{\varepsilon}{c} \sqrt{\frac{t}{\log(1/\delta)}} \cdot \frac{C_0C_1}{L \sqrt{k}} \cdot 2 \sqrt{t} \]

\[ \leq \frac{\varepsilon}{c} \sqrt{\frac{t}{\log(1/\delta)}}, \]

which finishes the induction. Now we have that

\[ \left\| Mx \right\|_2^2 - 1 \right\|_{L_t'} \leq \frac{\varepsilon}{c} \sqrt{\frac{t}{\log(1/\delta)}} \]

so we conclude that \( M \) has Strong \((\varepsilon, \delta)\)-JL Moment Property.

A simple corollary of this result is a sufficient condition for our recursive sketch \( \Pi^q \) to have the Strong JL Moment Property.

**Corollary 4.2. (Strong JL Moment for \( \Pi^q \))**

For any integer \( q \) which is a power of two, let \( \Pi^q : \mathbb{R}^m \rightarrow \mathbb{R}^m \) be defined as in Definition 3.3, where both of the common distributions \( S_{\text{base}} : \mathbb{R}^m \rightarrow \mathbb{R}^m \) and \( T_{\text{base}} : \mathbb{R}^d \rightarrow \mathbb{R}^m \), satisfy the Strong \((O(\frac{\varepsilon}{\sqrt{q}}), \delta)\)-JL Moment Property. Then it follows that \( \Pi^q \) satisfies the Strong \((\varepsilon, \delta)\)-JL Moment Property.

We conclude this section by proving Theorem 1.2.

**Proof of Theorem 1.2.** Let \( \delta = \Theta(\frac{\varepsilon^2}{\log(m)}) \) be the failure probability. Let \( \Pi^q \in \mathbb{R}^{m \times dp} \) be the sketch defined in Definition 3.3, where \( S_{\text{base}} \in \mathbb{R}^{m \times m^2} \) is a TensorSRHT sketch and \( T_{\text{base}} \in \mathbb{R}^{m \times d} \) is an OSNAP sketch with sparsity parameter \( s \), which will be set later.

Let \( m = \Theta\left(\frac{b \log(m)^2}{\varepsilon^2}\right) \) and \( s = \Theta\left(\sqrt{\frac{\log(n)}{\varepsilon}}\right) \) be integers, then Lemma 4.11 and Lemma 4.8 implies that \( S_{\text{base}} \) and \( T_{\text{base}} \) has the Strong \((O(\frac{\varepsilon}{\sqrt{q}}), \delta)\)-JL Moment Property, thus using Corollary 4.2 we conclude that \( \Pi^q \) has the Strong \((\varepsilon, \delta)\)-JL Moment Property and in particular it has the \((\varepsilon, \delta, \log(1/\delta))\)-JL Moment Property.

The correctness therefore follows from Lemma 4.12.

**Runtime of Algorithm 1.** We compute the time of running Algorithm 1 on a vector \( x \). Computing \( Y_j^0 \) for each \( j \) in lines 3 and 4 of algorithm requires applying an OSNAP sketch on either \( x \) or \( e_1 \) which takes time \( O(s \cdot m^2(x)) \). Therefore computing all \( Y_j^0 \)'s takes time \( O(qs \cdot m^2(x)) \). Computing each \( Y_j^1 \)'s for \( l \geq 1 \) in line 7 of Algorithm 1 amounts to
applying a TensorSRHT sketch of input dimension $m^2$ and target dimension $m$ on $Y_{2j}^{l-1} \otimes Y_{2j}^{l-1}$. This takes time $O(m \log m)$. Therefore computing $Y_j^l$ for all $l, j \geq 1$ takes time $O(q \cdot m \log m)$. Hence the total running time of Algorithm 1 on one vector $x$ is $O(pm \log_2 m + ps \cdot \text{nnz}(w))$. Sketching $n$ columns of a matrix $X \in \mathbb{R}^{d \times n}$ takes time $O(pnm \log_2 m + s \cdot \text{nnz}(X))$. In the setting of (1) we have that $s = O\left(\frac{\sqrt{s} \lambda \log^2 \frac{\delta}{\varepsilon}}{\varepsilon}\right)$, hence we get a runtime of $O\left(pnm + \frac{sp^2}{\varepsilon^2} \cdot \text{nnz}(X)\right)$. □

5 Linear Dependence on the Statistical Dimension $s_X$

In this section, we show that if one chooses the internal nodes and the leaves of our recursive construction from Section 3 to be TensorSRHT and OSNAP transform respectively, then the recursive construction $\Pi^q$ as in Definition 3.3 yields a high probability OSE with target dimension $O(p^4 s_X)$. Thus, we prove Theorem 1.3.

This sketch is very efficiently computable for high degree tensor products because the OSNAP transform is computable in input sparsity time and the TensorSRHT supports fast matrix vector multiplication for tensor inputs. We start by defining the Spectral Property for a sketch.

**Definition 5.1. (Spectral Property)** For any positive integers $m, n, d$ and any $\varepsilon, \delta \geq 0$, we say that a random matrix $S \in \mathbb{R}^{m \times d}$ satisfies the $(\mu_1, \mu_2, \varepsilon, \delta, n)$-spectral property if, for every fixed matrix $U \in \mathbb{R}^{d \times n}$ with $\|U\|^2_F \leq \mu_1$ and $\|U\|^2 \leq \mu_2$

\[
P \left[ \|U^T S^T U - U^T U\|_\text{op} \leq \varepsilon \right] \geq 1 - \delta.
\]

The spectral property is a central property of our sketch construction from Section 3 when leaves are OSNAP and internal nodes are TensorSRHT. This is a powerful property which implies that any sketch which satisfies the spectral property, is an Oblivious Subspace Embedding. The SRHT, TensorSRHT, as well as OSNAP sketches (Definitions 3.6, 3.7, 3.8 respectively) with target dimension $m = \Omega\left(\frac{\log^{\frac{3}{2}}(\frac{d}{\varepsilon})}{\varepsilon} \cdot \text{poly} \left(\log(\nu d/\delta)\right)\right)$ and sparsity parameter $s = \Omega(\text{poly} \left(\log(\nu d/\delta)\right))$, all satisfy the above-mentioned spectral property [Sar06, Tro11, NN13].

In Section 5.1 we recall the tools from the literature which we use to prove the spectral property for our construction $\Pi^q$. Then in Section 5.2 we show that our recursive construction in Section 3 satisfies the Spectral Property of Definition 5.1 as long as $I_{dv} \times T_{\text{base}}$ and $I_{m^2} \times S_{\text{base}}$ satisfy the Spectral Property. Therefore, we analyze the Spectral Property of $I_{dv} \times \text{OSNAP}$ and $I_{m^2} \times \text{TensorSRHT}$ in Section 5.3 and section 5.2 respectively. Finally we put everything together in Section 5.3 and prove that when the leaves are OSNAP and the internal nodes are TensorSRHT in our recursive construction of Section 3 the resulting sketch $\Pi^q$ satisfies the Spectral Property thereby proving Theorem 1.3.

5.1 Matrix Concentration Tools

In this section we present the definitions and tools which we use for proving concentration properties of random matrices.

**Claim 3.** For every $\epsilon, \delta > 0$ and any sketch $S \in \mathbb{R}^{m \times d}$ such that $I_k \times S$ satisfies $(\mu_1, \mu_2, \varepsilon, \delta, n)$-spectral property, the sketch $S \times I_k$ also satisfies the $(\mu_1, \mu_2, \varepsilon, \delta, n)$-spectral property.

We will use matrix Bernstein inequalities to show spectral guarantees for sketches (Theorem 6.1.1 of [Tro15]).

**Lemma 5.1. (Matrix Bernstein Inequality)** Consider a finite sequence $Z_i$ of independent, random matrices with dimensions $d_1 \times d_2$. Assume that each random matrix satisfies $\mathbb{E}[Z_i] = 0$ and $\|Z_i\|_{\text{op}} \leq B$ almost surely. Define $\sigma^2 = \max\{\|\sum_i \mathbb{E}[Z_i Z_i^*]\|_{\text{op}}, \|\sum_i \mathbb{E}[Z_i^* Z_i]\|_{\text{op}}\}$. Then for all $t > 0$,

\[
P \left[ \|\sum_i Z_i\|_{\text{op}} \geq t \right] \leq (d_1 + d_2) \cdot \exp \left(\frac{-t^2/2}{\sigma^2 + Bt/3}\right).
\]

**Lemma 5.2. (Corollary 6.2.1 of [Tro15])** Let $B$ be a fixed $n \times n$ matrix. Construct an $n \times n$ matrix $R$ that satisfies, $\mathbb{E}[R] = B$ and $\|R\|_{\text{op}} \leq L$ almost surely. Define $M = \max\{\|\mathbb{E}[RR^*]\|_{\text{op}}, \|\mathbb{E}[R^* R]\|_{\text{op}}\}$. Form the matrix sampling estimator, $\hat{R} = \frac{1}{m} \sum_{k=1}^m R_k$, where each $R_k$ is an independent copy of $R$. Then,

\[
P \left[ \|\hat{R} - B\|_{\text{op}} \geq \varepsilon \right] \leq 8n \cdot \exp \left(\frac{-m \varepsilon^2/2}{M + 2Lc/3}\right).
\]

To analyze the performance of SRHT we need the following claim which shows that with high probability individual entries of the Hadamard transform of a vector with random signs on its entries do not “overshoot the mean energy” by much.

**Claim 4.** Let $D$ be a $d \times d$ diagonal matrix with independent Rademacher random variables along the diagonal. Also, let $H$ be a $d \times d$ Hadamard matrix. Then, for every $x \in \mathbb{R}^d$,

\[
P_{D} \left[ \|HD \cdot x\|_{\infty} \leq 2\sqrt{\log_2 (d/\delta)} \cdot \|x\|_2 \right] \geq 1 - \delta.
\]
Claim 5. Let $D_1, D_2$ be two independent $d \times d$ diagonal matrices, each with diagonal entries given by independent Rademacher random variables. Also, let $H$ be a $d \times d$ Hadamard matrix. Then, for every $x \in \mathbb{R}^d$,

$$\Pr_{D_1, D_2} \left( \left\| (HD_1) - (HD_2) \right\|_\infty \leq 4 \log_2 \frac{d}{\delta} \right) \geq 1 - \delta.$$ 

5.2 Spectral Property of the sketch $\Pi^q$ In this section we show that the sketch $\Pi^q$ presented in Definition 5.1 inherits the spectral property (see Definition 5.1) from the base sketches $S_{\text{base}}$ and $T_{\text{base}}$. We start by the following claim which proves that composing two random matrices with spectral property results in a matrix with spectral property.

Claim 6. For every $\epsilon, \epsilon', \delta, \delta' > 0$, suppose that $S \in \mathbb{R}^{m \times t}$ is a sketch which satisfies the $((\mu_F + 1)(1 + \epsilon'), \mu_F + 1 + \epsilon', \epsilon, \delta, n)$-spectral property and also suppose that the sketch $T \in \mathbb{R}^{t \times d}$ satisfies the $(\mu_F + 1, \mu_F + 1, \epsilon', \delta'(1 + 1/n), n)$-spectral property. Then $S \cdot T$ satisfies the $(\mu_F + 1, \mu_F + 1 + \epsilon, \epsilon + \epsilon', \delta + \delta'(1 + 1/n), n)$-spectral property.

In the following lemma we show that composing independent random matrices with spectral property preserves the spectral property.

Lemma 5.3. For any $\epsilon, \delta, \mu_F, \mu > 0$ and every positive integers $k, n$, if $M^{(1)} \in \mathbb{R}^{d_1 \times d_1}, \ldots, M^{(k)} \in \mathbb{R}^{d_k \times d_k}$ are independent random matrices with the $(2\mu_F + 2, 2\mu_F + 2, O(\epsilon/k), O(\delta/nk), n)$-spectral property then the product matrix $M = M^{(k)} \cdots M^{(1)}$ satisfies the $(\mu_F + 1, \mu_F + 1, \epsilon, \delta, n)$-spectral property.

The following lemma shows that our sketch construction $\Pi^q$ presented in Definition 3.3 inherits the spectral property of Definition 5.1 from the base sketches, that is, if $S_{\text{base}}$ and $T_{\text{base}}$ are such that $I_{m-n} \times S_{\text{base}}$ and $I_{d-k} \times T_{\text{base}}$ satisfy the spectral property, then the sketch $\Pi^q$ satisfies the spectral property.

Lemma 5.4. For every positive integers $n, d, m, q$ of two integer $q$, any base sketch $T_{\text{base}} : \mathbb{R}^d \rightarrow \mathbb{R}^m$ such that $T_{\text{base}} \times \mathbb{R}^n$ satisfies the $(2\mu_F + 2, 2\mu_F + 2, O(\epsilon/q), O(\delta/nq), n)$-spectral property, any $S_{\text{base}} : \mathbb{R}^m \rightarrow \mathbb{R}^n$ such that $I_{m-n} \times S_{\text{base}}$ satisfies the $(2\mu_F + 2, 2\mu_F + 2, O(\epsilon/q), O(\delta/nq), n)$-spectral property, the sketch $\Pi^q$ defined as in Definition 3.3 satisfies the $(\mu_F + 1, \mu_F + 1, \epsilon, \delta, n)$-spectral property.

Spectral Property of Identity $\times$ TensorSRHT In next lemma, we show that tensoring an identity operator with a TensorSRHT sketch results in a transform that satisfies the spectral property defined in Definition 5.1 with nearly optimal target dimension.

Lemma 5.5. Suppose $\epsilon, \delta, \mu_F > 0$ and $n$ is a positive integer. If $m = \Omega(\log(n/d)\log^2(n/d)), S \in \mathbb{R}^{m \times d}$ is a TensorSRHT sketch, then the sketch $I_k \times S$ satisfies $(\mu_F, \mu_F, \epsilon, \delta, n)$-spectral property.

Spectral Property of Identity $\times$ OSNAP In next lemma, we show that tensoring identity operator with OSNAP sketch (Definition 3.8) results in a transform which satisfies the spectral property (Definition 5.1) with small target dimension as well as fast application time for sketching sparse vectors.

Lemma 5.6. Suppose $\epsilon, \delta, \mu_F > 0$ and $n$ is a positive integer. If $S \in \mathbb{R}^{m \times d}$ is a OSNAP sketch with sparsity parameter $s$, then the sketch $I_k \times S$ satisfies the $(\mu_F, \mu_F, \epsilon, \delta, n)$-spectral property, provided that $s = \Omega\left(\frac{\log(2n/d)\log(n/d)}{\epsilon^3 n}\right)$ and $m = \Omega\left(\frac{(\mu_F^2/\epsilon^2)\cdot \log^2(n/d)}{\delta^2}\right)$.

5.3 High Probability OSE with linear dependence on $s$ We are ready to prove Theorem 1.3. We prove that if we instantiate $\Pi^q$ from Definition 3.3 with $T_{\text{base}} : \text{OSNAP}$ and $S_{\text{base}} : \text{TensorSRHT}$, it satisfies the statement of Theorem 1.3.

Proof of Theorem 1.3. Let $\delta = \frac{1}{\text{poly}(n)}$ denote the failure probability. Let $m = p^3\log_2(\frac{nd}{\delta}) \cdot \frac{\epsilon}{\varepsilon}$ and $s = \frac{p^3}{\varepsilon^2} \cdot \log_2(\frac{nd}{\delta})$ be integers. Let $\Pi^q \in \mathbb{R}^{m \times m}$ be the sketch defined in Definition 3.3 where $S_{\text{base}} \in \mathbb{R}^{m \times m^2}$ is a TensorSRHT sketch and $T_{\text{base}} \in \mathbb{R}^{n \times d}$ is an OSNAP sketch with sparsity parameter $s$.

Let $q = 2\lceil \log_2(p) \rceil$. It is sufficient to show that $\Pi^q$ is a $(\epsilon, \delta, s_\lambda, d^2, n)$-Oblivious Subspace Embedding. Consider arbitrary $A \in \mathbb{R}^{d^2 \times n}$ and $\lambda > 0$. We let denote the statistical dimension of $A$ by $s_\lambda = s_\lambda(A^T A)$. Let $U = A(A^T A + \lambda I_n)^{-1/2}$. Therefore, $\|U\|_2^2 \leq 1$ and $\|U\|_F^2 = s_\lambda$. Since $q < 2p$, by Lemma 5.6 the transform $I_{n-q} \times T_{\text{base}}$, satisfies $(2s_\lambda + 2, O(\epsilon/q), O(\delta/nq), n)$-spectral property. Moreover, by Lemma 5.5, the transform $I_{m-n} \times S_{\text{base}}$ satisfies $(5s_\lambda + 9, 9, O(\epsilon/q), O(\delta/nq^2), n)$-spectral property. Therefore, by Lemma 5.4 the sketch $\Pi^q$ satisfies $(s_\lambda, 1, 1, \epsilon, \delta, n)$-spectral property, hence,

$$\Pr[\left\| (\Pi^q U)^\top \Pi^q U - U^\top U \right\|_{op} \leq \epsilon] \geq 1 - \delta.$$

Since $U^\top U = (A^T A + \lambda I_n)^{-1/2} A^T A (A^T A + \lambda I_n)^{-1/2}$ and $\Pi^q U = \Pi^q A(A^T A + \lambda I_n)^{-1/2}$ the following holds with probability at least $1 - \delta$,

$$A^T A + \lambda I_n \preceq (\Pi^q A)^\top \Pi^q A + \lambda I_n \preceq A^T A + \frac{\lambda I_n}{1 - \epsilon}.$$

Runtime: By Lemma 3.1 the sketched matrix $\Pi^q A$ can be computed using Algorithm 1. When $S_{\text{base}}$
is TensorSRHT and its base is OSNAP, the runtime of Algorithm 1 for a fixed vector \( w \in \mathbb{R}^d \) is as follows; Computing \( Y_j^{(1)} \)'s for each \( j \) in lines 3 and 4 of algorithm requires applying an OSNAP sketch on \( w \in \mathbb{R}^d \) which on expectation takes time \( O(s \cdot \text{nnz}(w)) \). Therefore computing all \( Y_j^{(1)} \)'s takes time \( O(qs \cdot \text{nnz}(w)) \). Computing each of the terms \( Y_j^{(1)} \)'s in line 7 of algorithm amounts to applying a TensorSRHT of input dimension \( m_1^2 \) and target dimension of \( m \) on \( Y_j^{t;l-1} \). This takes time \( O(m \log m) \). Therefore computing all the \( Y_j^{(1)} \)'s takes \( O(g \cdot m \log m) \). Hence the total runtime of Algorithm 1 on a vector \( w \) is \( O(p \cdot m \log_2 m + ps \cdot \text{nnz}(w)) \). Therefore, sketching \( n \) columns of a matrix \( X \in \mathbb{R}^{d \times n} \) takes time \( O(p(nm \log_2 m + s \cdot \text{nnz}(X))) \). □

6 Oblivious Subspace Embedding for the Gaussian Kernel

In this section we show how to sketch the Gaussian kernel matrix by polynomial expansion and then applying our proposed sketch for the polynomial kernels.

Proof of Theorem 1.4 Let \( \delta = \frac{1}{\text{poly}(n)} \) denote the failure probability. Note that \( G_{i,j} = e^{-\|x_i\|^2/2} \cdot e^{x_i \top x_j} \), \( e^{-\|x_i\|^2/2} \) for every \( i, j \in [n] \). Let \( D \) be a \( n \times n \) diagonal matrix with \( l \)th diagonal entry \( e^{-\|x_i\|^2/2} \) and let \( K \in \mathbb{R}^{n \times n} \) be defined as \( K_{i,j} = e^{x_i \top x_j} \) (note that \( DKD = G \)). we remark that \( K \) is a psd matrix. The Taylor series expansion for kernel \( K \) is as follows,

\[
K = \sum_{l=0}^{\infty} \frac{(X \otimes l) \top X \otimes l}{l!}.
\]

Therefore \( G \) can be written as the following series,

\[
G = \sum_{l=0}^{\infty} \frac{(X \otimes l) \top X \otimes l}{l!} D \sum_{l=0}^{\infty} \Pi^l.
\]

Note that each of the terms \((X \otimes l) \top X \otimes l \mid l \geq 0\) is upper bounded by that of \( G \) through the following claim.

Claim 7. For every \( \mu \geq 0 \) and every integer \( l \),

\[
s_\mu ((X \otimes l) \top X \otimes l) \leq s_\mu(G).
\]

This claim can be proved by Courant-Fischer’s min-max theorem. If we let \( P = \sum_{l=0}^{q} \frac{(X \otimes l) \top X \otimes l}{l!} \), where \( q = C \cdot (r^2 + \log(\frac{1}{\delta})) \) for some constant \( C \), then by the triangle inequality we have

\[
\|K - P\|_{\text{op}} \leq \sum_{l=q}^{\tau} \frac{\|X \otimes l\|_{\text{op}}}{l!} \leq \sum_{l=q}^{\tau} \frac{\|X \otimes l\|_{\text{op}}}{l!} \leq \epsilon \lambda/2.
\]

\( P \) is a psd matrix. Also note that all the eigenvalues of the diagonal matrix \( D \) are bounded by 1. Hence, in order to get a subspace embedding it is sufficient to satisfy the following with probability \( 1 - \delta \),

\[
\frac{DPD + \lambda I_n}{1 + \epsilon/2} \leq (S_g(X))^\top S_g(X) + \lambda I_n \leq \frac{DPD + \lambda I_n}{1 - \epsilon/2}.
\]

Let \( \Pi^l \in \mathbb{R}^{m \times d} \) be the sketch from Theorem 1.3 By Claim 7 we get the following guarantee on \( \Pi^l \) with probability at least \( 1 - \frac{\delta}{d^\ell} \) as long as \( m_l = \Omega \left( l^4 \log^3 (nd/\delta) \cdot s_\lambda / \epsilon^2 \right) \)

\[
\frac{A^\top A + \lambda I}{1 + \epsilon/2} \leq (\Pi^l A)^\top \Pi^l A + \lambda I \leq \frac{A^\top A + \lambda I}{1 - \epsilon/2},
\]

where \( A = X \otimes l \). Moreover, \( \Pi^l X \otimes l D \) can be computed using \( O \left( n \cdot l \cdot m_l \log_2 m + \frac{m_l}{4} \cdot \log^3 (nd/\delta) \cdot \text{nnz}(X) \right) \) runtime where \( s_\lambda \) is the \( \lambda \)-statistical dimension of \( G \).

We let \( S_P \) be the sketch of size \( m \times (\sum_{l=0}^{q} d^l) \) which sketches the kernel \( P \), defined as

\[
S_P = \frac{1}{\sqrt{q!}} \Pi^0 \oplus \frac{1}{\sqrt{q!}} \Pi^1 \oplus \frac{1}{\sqrt{q!}} \Pi^2 \cdots \frac{1}{\sqrt{q!}} \Pi^q.
\]

Let \( Z \) be the matrix of size \( (\sum_{l=0}^{q} d^l) \times n \) whose \( i \)th column is \( z_i = x_i \otimes 0 \oplus x_i \otimes 1 \oplus x_i \otimes 2 \cdots x_i \otimes q \), where \( x_i \) is the \( i \)th column of \( X \). Therefore the following holds for \((S_P Z)^\top S_P Z\),

\[
(S_P Z)^\top S_P Z = \sum_{l=0}^{q} \frac{\Pi^l X \otimes l}{l!},
\]

and hence, \((S_P Z D)^\top S_P Z D = \sum_{l=0}^{q} \frac{\Pi^l X \otimes l}{l!} \Pi^l X \otimes l D \) Therefore by combining the terms of (6.7) for all \( 0 \leq l \leq q \), using a union bound we get that with probability \( 1 - \delta \), the following holds,

\[
\frac{DPD + \lambda I_n}{1 + \epsilon/2} \leq (S_g Z D)^\top S_g Z D + \lambda I_n \leq \frac{DPD + \lambda I_n}{1 - \epsilon/2}.
\]

Now we define \( S_g(x) \) which is a non-linear transformation on the input \( x \) defined as

\[
S_g(x) = e^{-\|x\|^2/2} \bigoplus_{l=0}^{q} \frac{1}{\sqrt{l!}} \cdot \Pi^l (x \otimes l).
\]
We have that $S_g(X) = S_{PZD}$, therefore with probability $1 - \delta$, the following holds,

$$\frac{G + \lambda I_n}{1 + \varepsilon} \preceq (S_g(X))^\top S_g(X) + \lambda I_n \preceq \frac{G + \lambda I_n}{1 - \varepsilon}.$$ 

Note that the target dimension of $S_g$ is

$$m = m_0 + m_1 + \cdots + m_q \approx q^5 \log^3(n \delta) s_\lambda / \varepsilon^2.$$ 

Also, by Theorem 1.3, time to compute $S_g(X)$ is

$$O \left( \frac{n^2}{\varepsilon} \cdot \log^4(n \delta) \cdot s_\lambda + \frac{n^3}{\varepsilon} \cdot \log^3(n \delta) \cdot \text{nnz}(X) \right).$$

\[\square\]

7 Acknowledgements

Anders Aamand for early discussions of orthogonality and moments.

References


